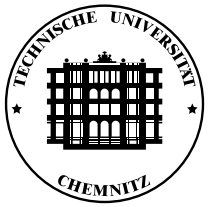

TU Chemnitz
Fakultät für Informatik
Professur Rechnernetze und verteilte Systeme

Diplomarbeit

Zugangsmanagement für Wireless LAN

Eingereicht von: Mirko Parthey
Wissenschaftlicher Betreuer: Prof. Dr.-Ing. habil. Uwe Hübner
Betreuer im URZ: Günther Fischer
Beginn der Arbeit: 1. April 2001
Abgabedatum: 31. Dezember 2001

Aufgabenstellung

Zugangsmanagement für Wireless LAN

Realisieren Sie einen transparenten Zugang zum Campusnetz der TU Chemnitz und zum Internet für mobile Geräte. Dabei sollen verschiedene Nutzerklassen unterschieden und Sicherheitsprobleme berücksichtigt werden. Die Notwendigkeit von Eingriffen in die mobilen Geräte soll möglichst gering sein.

Inhaltsverzeichnis

1 Einführung **8**

2 Lösungsvarianten **10**
2.1 Anforderungen 10
2.2 Authentifizierung durch Hardware-Identität 11
2.3 Link-Verschlüsselung 11
2.4 Nutzerbasierte Authentifizierung 11
2.5 Nutzerbasierte Auth. + Verschlüsselung 12

3 Grundlagen **13**
3.1 Einordnung 13
3.2 Firewalls 14
3.3 Netfilter (*iptables*) 15
3.4 Common Gateway Interface (CGI) 16
3.5 Pluggable Authentication Modules (PAM) 16
3.6 Kerberos 17
3.7 Dynamic Host Configuration Protocol (DHCP) 17
3.8 Domain Name System (DNS) 18
3.9 Internet Control Message Protocol (ICMP) 18
3.10 Address Resolution Protocol (ARP) 19

4 Praxis **20**
4.1 Einordnung 20
4.2 Komponenten 21
4.2.1 HTTP-Server 21
4.2.2 DHCP-Server 21
4.2.3 DNS-Server 21
4.2.4 Skripte 22
4.2.5 Firewall-Regeln 23
4.3 Funktionsweise 23
4.3.1 Registrierung eines Clienten 24
4.3.2 Erreichbarkeitsprüfung und Deregistrierung 26
4.3.3 Firewall-Regeln und Nutzerklassen 27

4.3.4 ZIN-Informationen 29
4.4 Entwurfsentscheidungen 30
4.4.1 Ähnliche Software als Grundlage 30
4.4.2 Standardsoftware 31
4.4.3 Programmiersprache 32
4.4.4 Authentifizierung 32
4.4.5 Firewall-Regeln 33
4.4.6 Funktionsweise 34
4.4.7 Sicherheitsfragen 35
4.4.8 Prozesse und Interprozeßkommunikation 36
4.4.9 Performance und Optimierungsmöglichkeiten 36
4.5 Administration 37
4.5.1 Systemvoraussetzungen und Installation 38
4.5.2 Systemstart und Herunterfahren 38
4.5.3 Dateien 39
4.5.4 Standardsoftware und deren Konfiguration 43

5 Fazit und Perspektiven **49**
5.1 Bewertung 49
5.2 Entwicklungsmöglichkeiten 50
5.2.1 Gastzugang mit erweiterten Rechten 50
5.2.2 Unterstützung für mehrere Funk-Subnetze 50
5.2.3 Mobile IP 50

A Administration: HOWTO **52**
A.1 Änderung von IP-Adressen oder Subnetzen 52
A.1.1 Firewall-IP auf dem externen Interface ändern 52
A.1.2 Firewall-IP auf dem internen Interface ändern 52
A.1.3 Subnetz der offiziellen IP-Adressen ändern 53
A.1.4 Subnetz der privaten IP-Adressen ändern 53
A.2 Hinzufügen von Nutzerklassen 54
A.3 Anmeldung von Clienten der Nutzerklasse `static` 55

B Nutzerdokumentation **56**
B.1 Systemanforderungen 56
B.2 Ablauf der Anmeldung 57

Abbildungsverzeichnis

3.1 Anordnung der Netfilter-Chains 15

4.1 Funknetz mit Firewall 20
4.2 Struktur der Firewall-Software 23

B.1 Anmeldeformular 57

Tabellenverzeichnis

1.1 Funk-Technologien zur Datenübertragung 8

3.1 Einordnung ins OSI-Schichtenmodell 13

4.1 Allgemeingültige Firewall-Variablen 41
4.2 Clientspezifische Firewall-Variablen 42
4.3 Bedeutung der Spalten in der Datei `clients` 42

B.1 DHCP-Clienten im Test 56

Kapitel 1

Einführung

Das Internet hat in den letzten Jahren sein exponentielles Wachstum fortgesetzt [1] und ist inzwischen zu einem Massenmedium geworden. Eine weitere herausragende technische Entwicklung der letzten Zeit ist die breite Verfügbarkeit von mobilen Rechnern und Mobiltelefonen.

Wenn man diese Möglichkeiten in Verbindung betrachtet, ist der Wunsch nach einem Internetzugang für mobile Geräte naheliegend. Um die Anbindung komfortabel zu gestalten, ist eine drahtlose Zugangstechnologie erforderlich, zum Beispiel Infrarot oder Funk, wobei Infrarot allerdings nur für kurze Strecken mit Sichtkontakt geeignet ist. Einige zur Datenübertragung geeignete Funktechnologien sind in Tabelle 1.1 aufgeführt.

Technologie	**Abk.**	**Referenz**
Global System for Mobile Communications	GSM	[2][3]
Universal Mobile Telecommunications System	UMTS	[3][4]
Digital Enhanced Cordless Telecommunication	DECT	[5][6]
Wireless Local Area Network	WLAN	[7][9]
Bluetooth		[10][11]

Tabelle 1.1: Funk-Technologien zur Datenübertragung

Im Rahmen einer Förderinitiative des Bundesministeriums für Bildung und Forschung (BMBF) [12] wurde an der Technischen Universität Chemnitz eine Infrastruktur für Wireless LAN nach IEEE 802.11b [8] installiert. Für den mobilen Netzzugang in Gebäuden bietet Wireless LAN folgende Vorteile gegenüber anderen Funktechnologien:

- vergleichsweise hohe Bandbreite
- vertretbare Kosten für Infrastruktur und Ausrüstung mobiler Geräte
- einfache Integration in bestehende Netzwerke auf Ethernet-Basis

Bei der Einrichtung eines drahtlosen Netzzugangs in öffentlichen Gebäuden müssen allerdings auch Sicherheitsfragen berücksichtigt werden. Eine Besonderheit von Funkverbindungen liegt darin, daß das Übertragungsmedium von allen Teilnehmern gemeinsam genutzt wird. Deshalb ist das Abhören fremder Datenströme prinzipiell möglich, und außerdem kann der Zugang zum Übertragungsmedium nicht so einfach gesteuert werden, wie es bei der Freischaltung von Ethernet-Anschlußdosen möglich ist.

Für diese beiden Probleme gibt es bereits verschiedene Lösungen. Manche davon sind allerdings zu aufwendig für Administratoren oder Nutzer, andere haben sich außerdem als unsicher erwiesen, wie z. B. Wired Equivalent Privacy (WEP) [13].

Die vorliegende Arbeit beschreibt eine Zugangsmanagement-Lösung, die nicht auf WEP basiert, geringen Verwaltungsaufwand erfordert und differenzierte Zugangsrechte in Form von Nutzerklassen ermöglicht. Auf Nutzerseite ist dafür nur Software erforderlich, die auf Rechnern mit Netzzugang üblicherweise bereits installiert ist. Das Problem des Abhörens fremder Daten wird im Rahmen dieser Arbeit nicht gelöst. Den Nutzern wird stattdessen empfohlen, die Übertragung selbst zu sichern, z.B. mit SSH [15][16], SSL/TLS [17][18] oder IPsec [19][20].

Kapitel 2

Lösungsvarianten

2.1 Anforderungen

Der Netzwerkzugang per Wireless LAN bietet die Möglichkeit, einen eigenen Rechner drahtlos ans Campusnetz anzuschließen, wobei dieser Anschluß eine vergleichsweise hohe Bandbreite bietet. Die Nutzung dieser Ressourcen ist zwar einfach, erfordert aber einen verantwortungsvollen Umgang, der besonders bei neuen Nutzern nicht ohne weiteres vorausgesetzt werden kann. Mögliche Probleme sind die Übertragung übermäßig großer Datenvolumina, Angriffe durch den Nutzer auf Internet-Rechner und Angriffe aus dem Internet auf den Nutzerrechner sowie die Infektion des Nutzerrechners mit Viren oder Trojanischen Pferden.

Mit einem selbst administrierten Rechner kann ein Nutzer größere Störungen verursachen als z. B. mit den öffentlichen Rechnern des Universitätsrechenzentrums (URZ). Deshalb benötigen alle Nutzer dieses Dienstes ein *Zertifikat Internet-Nutzung (ZIN)* [24], um zumindest die Probleme zu verringern, die durch Unwissenheit oder Sorglosigkeit verursacht werden. Ohne dieses Zertifikat soll nur ein eingeschränkter Gastzugang möglich sein.

Um Anschluß an ein WLAN zu erhalten, ist kein physischer Zugang zu einem Anschlußpunkt erforderlich, der Zugriff auf das Netzwerk kann sogar von außerhalb des Gebäudes noch möglich sein. Deshalb ist es auch für Außenstehende relativ einfach, das Netzwerk unberechtigt zu nutzen. Um das zu verhindern, definiert IEEE 802.11 [7] einen Authentifizierungsmechanismus in der Verbindungsschicht, außerdem bieten manche Geräte noch herstellerspezifische Authentifizierungsmöglichkeiten. In [21] wird allerdings gezeigt, daß diese Authentifizierung gefälscht werden kann und damit nur einen geringen Nutzen hat.

Sowohl die Prüfung, ob der Nutzer ein ZIN besitzt, als auch der Schutz vor unberechtigter Nutzung erfordern eine Authentifizierung. Die nächsten Abschnitte beschreiben einige Varianten, wie diese ablaufen könnte, sowie die Vor- und Nachteile.

2.2 Authentifizierung auf Basis der Hardware-Identität

Die in der TU Chemnitz installierten Access Points (Orinoco AP-1000) bieten die Möglichkeit, nur Clienten mit einer bestimmten Hardware-Identität (Ethernet-MAC-Adresse) zuzulassen. Dafür ist es erforderlich, die Netzwerkkarten aller berechtigten Nutzer zentral zu registrieren. Die Authentifizierung erfolgt dabei manuell, z. B. durch Vorlage des Personalausweises. Das bedeutet einen erheblichen Aufwand und kann Probleme mit sich bringen, wenn ein Nutzer eine bereits registrierte Netzwerkkarte weitergibt. Um eine Datenübertragung nachträglich einem Nutzer zuordnen zu können, ist außerdem der Einsatz von statischem ARP bzw. eine Protokollierung der ARP-Tabelle auf dem Router erforderlich.

2.3 Link-Verschlüsselung

Für die Verschlüsselung auf der Verbindungsschicht ist in [7] das Protokoll *Wired Equivalent Privacy (WEP)* definiert. WEP bietet symmetrische Verschlüsselung mit maximal 4 konfigurierbaren Schlüsseln. Es definiert keinen Mechanismus für das Schlüsselmanagement. Bei den hier eingesetzten WLAN-Netzwerkkarten werden die Schlüssel nicht dauerhaft auf der Karte gespeichert, sondern der Treiber muß sie jedesmal bei der Initialisierung auf die Karte übertragen. Deshalb müssen die Schlüssel in mehr oder weniger lesbarer Form außerhalb der Karte vorliegen. Unter Windows werden sie verschleiert in der Registry gespeichert, unter Linux liegen sie standardmäßig im Klartext vor. Damit sind die Schlüssel in jedem Fall kopierbar, und es ist nicht sichergestellt, daß nur Berechtigte Zugang zum Netzwerk erhalten.

Aufgrund der begrenzten Zahl an Schlüsseln würden viele Nutzer den gleichen Schlüssel erhalten. Wenn die Berechtigung eines Nutzers abläuft, müßte dieser Schlüssel geändert und allen anderen Nutzern wieder mitgeteilt werden. Das ist nicht praktikabel.

In [13] wird gezeigt, daß die Ermittlung eines WEP-Schlüssels durch einen passiven Angriff (Abhören) in vertretbarer Zeit möglich ist. Deswegen wird dort empfohlen, sich nicht auf die Sicherheitsmechanismen der Verbindungsschicht zu verlassen.

2.4 Nutzerbasierte Authentifizierung

Die zu entwickelnde Software soll möglichst keine Eingriffe auf dem Nutzerrechner erfordern, das gilt auch für die Authentifizierung. Ein Protokoll, das diese Anforderung erfüllt und außerdem als sicher gilt, ist die HTTP-Authentifizierung über SSL/TLS. Der Nutzer meldet sich für jede Sitzung

mit Nutzerkennzeichen und Paßwort an, dabei werden seine MAC- und IP-Adresse zeitweilig registriert. Während der Sitzung erfolgt nur eine schwache Authentifizierung auf Basis dieser Daten. Die Erreichbarkeit des Nutzerrechners wird regelmäßig geprüft und nach einer gewissen Zeit der Nichterreichbarkeit wird die Registrierung wieder gelöscht. Damit ist die unberechtigte Nutzung eines registrierten Zugangs deutlich erschwert.

Eine ähnliche Möglichkeit stellt *Port Based Network Access Control* nach IEEE 802.1X [14] dar, es arbeitet aber auf der Verbindungsschicht. Ein Client, der den Netzzugang nutzen möchte, hat zunächst nur Zugang zur Bridge (Access Point), um sich zu authentifizieren. Der Access Point kann sich dann an einen Authentifizierungsserver wenden, um die Daten zu prüfen. War die Authentifizierung erfolgreich, so wird der Port freigeschaltet. Beim Trennen der Verbindung wird die Freischaltung wieder aufgehoben.

Im Kontext von Wireless LANs gilt als Verbindung die Assoziation zwischen Mobilrechner und Access Point, die Ports existieren nur als logische Ports innerhalb des Access Points. Um 802.1X einzusetzen, ist Unterstützung sowohl durch die Access Points als auch durch die Mobilrechner nötig. Die hier eingesetzten Access Points bieten diese Unterstützung nicht, ebensowenig ist die entsprechende Forderung an die Mobilrechner vertretbar.

2.5 Nutzerbasierte Authentifizierung und Verschlüsselung der Daten

In Funknetzwerken ist eine zusätzliche Verschlüsselung und Authentifizierung für die übertragenen Daten wünschenswert. Das läßt sich z. B. mit dem *Point-to-Point-Tunneling-Protokoll (PPTP)*, *IP Security (IPsec)* oder einer Kombination von *Extensible Authentication Protocol (EAP)* [23] und WEP realisieren. Bei der letztgenannten Möglichkeit werden die WEP-Schlüssel dynamisch vergeben, so daß der Angriff aus [13] geringfügig erschwert ist. Allerdings bieten die hier eingesetzten Access Points keine Unterstützung für dieses Protokoll.

PPTP erfüllt die Forderung nach einfacher Nutzung. Es ermöglicht aber Wörterbuchangriffe, weil es den Schlüssel aus dem Paßwort ableitet, und ist deshalb unsicher. [22]

Für IPsec ist meist zusätzliche Software auf dem Nutzerrechner erforderlich. Außerdem muß der öffentliche Schlüssel jedes Nutzerrechners auf einem sicheren Weg ins Rechenzentrum übermittelt werden. Der Verwaltungsaufwand entspricht in etwa der ersten Variante (Registrierung der MAC-Adressen). Verglichen mit allen anderen genannten Varianten bietet IPsec die höchste Sicherheit.

Als Kompromiß zwischen einfacher Benutzbarkeit und ausreichender Sicherheit fiel in dieser Arbeit die Wahl auf die nutzerbasierte Authentifizierung über HTTPS ohne besonderen Schutz der später übertragenen Nutzdaten.

Kapitel 3

Grundlagen

3.1 Einordnung

Bei der Lösungsbeschreibung werden verschiedene Übertragungsprotokolle erwähnt. Tabelle 3.1 gibt einen Überblick, wie sich diese Protokolle ins OSI-Schichtenmodell einordnen.

Schicht		**Protokoll**	**RFC**
7	Anwendung	HTTP DHCP DNS	2616 2131 1034, 1035
6	Präsentation		
5	Sitzung		
4	Transport	TCP UDP SSL/TLS	793 768 2246
3	Netzwerk	IP ICMP Mobile IP	791 792 2002
2	Verbindung	PPTP ARP	2637 826
1	physikalisch		

Tabelle 3.1: Einordnung ins OSI-Schichtenmodell

Um den Zugriff der Mobilrechner auf das Campusnetz und das Internet steuern zu können, ist ein Firewall erforderlich. Meist werden Firewalls zwischen dem Netz einer Einrichtung und dem übrigen Internet eingesetzt. Im vorliegenden Fall dient der Firewall aber zum Trennen von Sicherheitsdomänen innerhalb einer Organisation. [25]

3.2 Firewalls

Man kann Firewalls auf 2 Arten realisieren: als *Paketfilter* und *Proxy Server*. Ein *Paketfilter* arbeitet auf der Netzwerkschicht, er wertet für jedes einzelne IP-Datagramm verschiedene Parameter aus, wie z.B. Quell- und Zieladresse. Anhand dieser Merkmale trifft er eine Entscheidung, ob das Paket weitergeleitet, verworfen oder mit einer Fehlermeldung zurückgewiesen wird. Manche Paketfilter bieten zusätzlich noch die Funktion, bei der Weiterleitung die Quell- oder Zieladresse des Paketes zu ändern.

Viele Paketfilter können auch Informationen aus der Verbindungsschicht in diese Entscheidung mit einbeziehen. Solange das nur Informationen sind, die in jedem Paket einer Übertragung vorhanden sind (Portnummern, IDs), dann handelt es sich um einen *zustandslosen* Paketfilter. Wenn dagegen der Firewall Informationen speichert, um später eintreffende Pakete einer früheren Übertragung zuordnen zu können (TCP-Verbindungen, ICMP echo request/reply), dann bezeichnet man ihn als *zustandsbehaftet*.

Ein *Proxy Server* nimmt eine Anfrage von einem Clienten entgegen, leitet diese Anfrage an den eigentlichen Empfänger weiter, und überträgt dann das Ergebnis wieder an den ursprünglichen Absender der Anfrage. Proxies arbeiten in der Verbindungsschicht (SOCKS Proxy) oder der Anwendungsschicht (HTTP Proxy). [26]

Ein SOCKS Proxy arbeitet ähnlich wie eine (alte) Telefonvermittlung. Er nimmt eine TCP-Verbindung vom Client an und baut daraufhin selbst eine TCP-Verbindung zum Zielsystem auf. Zwischen diesen beiden Verbindungen leitet er die Daten weiter. SOCKS Proxies bieten keine Nutzerauthentifizierung und haben deshalb kaum Vorteile gegenüber einem zustandsbehafteten Paketfilter, sie sind sogar unflexibler.

Ein Anwendungsproxy nimmt eine Anfrage vom Client entgegen, interpretiert die Informationen des Anwendungsprotokolls und stellt daraufhin eine (eventuell modifizierte) Anfrage an den eigentlichen Empfänger. Manche Anwendungsproxies arbeiten auch als Zwischenspeicher *(cache)*, um bei wiederholtem Abruf der gleichen Daten die Zugriffszeit und das übertragene Datenvolumen zu verringern. Wenn die angeforderten Daten bereits im Zwischenspeicher vorliegen, brauchen sie die Anfrage nicht weiterzuleiten, sondern können sie selbst beantworten.

Anwendungsproxies ermöglichen eine feiner abgestufte Kontrolle als Paketfilter. Z. B. kann man sie benutzen, um für bestimmte Dienste eine Nutzerauthentifizierung zu erzwingen. Allerdings erfordert ihr Einsatz auch einen hohen Aufwand, weil für jedes Anwendungsprotokoll ein eigener Proxy-Dienst erforderlich ist, und nicht für alle Protokolle sind überhaupt Proxies verfügbar.

Ein genereller Nachteil von Proxies gegenüber Paketfiltern besteht darin, daß die Nutzung des Proxies am Client zu konfigurieren ist und dieser Proxies unterstützen muß. Nutzt man eine automatische Umleitung mit Hilfe

von Paketfiltern, dann kann diese Anforderung aber entfallen.

Um einen Proxy als Sicherheitseinrichtung zu verwenden, muß er der einzige Weg sein, auf dem die Daten eines bestimmten Protokolls übertragen werden können. Solange die gesamte Kommunikation zwischen den Sicherheitsdomänen über den Proxy ablaufen soll, genügt es, auf dem Proxy-Rechner das Routing zu deaktivieren. Anderenfalls bietet sich wieder die Kombination mit einem Paketfilter an.

3.3 Netfilter (*iptables*)

Netfilter implementiert einen Paketfilter und *Network Address Translation (NAT)* für die Linux-Kernel-Serie 2.4. Die Datenpakete durchlaufen dabei eine Folge von *chains*, innerhalb derer sie von Netfilter bearbeitet werden.

Durch die Filterfunktion können Pakete weitergeleitet, zurückgewiesen oder verworfen werden. Mit NAT können IP-Adressen und Portnummern geändert werden. Es ist außerdem möglich, im Paket Datenfelder wie *Time-to-Live (TTL)* oder *Type of Service (TOS)* zu ändern (*mangle*). Jede der Chains ist für eine eigene Art der Behandlung (Filter, NAT, mangle) zuständig, bzw. sie erhält die Pakete in einem anderen Stadium der Bearbeitung durch den Netzwerk-Code, siehe Abbildung 3.1 oder [27] [28].

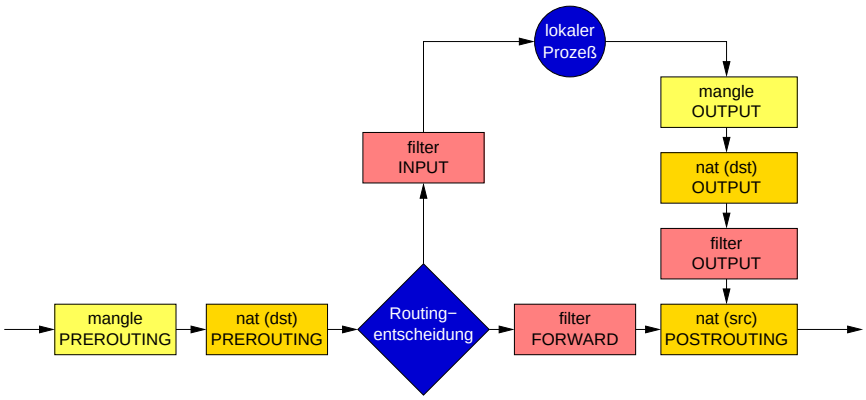

Abbildung 3.1: Anordnung der Netfilter-Chains

Als Entscheidungskriterium, wie mit den Paketen zu verfahren ist, können u. a. Ethernet-MAC-Adressen, IP-Adressen oder Portnummern der Verbindungsschicht dienen. Die Auswertung kann auch zweistufig erfolgen: Zuerst wird das Paket nach den genannten Kriterien klassifiziert und entsprechend gekennzeichnet (*firewall mark*). In den Chains, die das Paket danach durchläuft, kann man diese Kennzeichnung dann wieder verwenden, um über seine weitere Behandlung zu entscheiden.

Mit *NAT* ist es möglich, die Quell- oder Zieladresse von Paketen zu

ändern. Das Ändern von Zieladressen (*Destination NAT, DNAT*) kann zur Lastverteilung, zur Realisierung eines transparenten Proxys oder zum Umleiten von Daten auf einen nicht direkt erreichbaren Zielrechner dienen. Eine Änderung der Quelladresse (*Source NAT, SNAT*) ist z. B. sinnvoll, wenn diese aus einem privaten Adreßbereich stammt [29]. Die Antwortpakete werden ebenfalls von NAT behandelt, um die ursprünglichen Adressen wiederherzustellen.

Wenn der Adreßraum komprimiert wird (z. B. bei n:1 NAT), muß trotzdem eine umkehrbar eindeutige Adreßzuordnung möglich sein. Zu diesem Zweck ändert NAT Adreßinformationen in der Verbindungsschicht, z. B. die Portnummer. Ist die IP-Adreßzuordnung bereits umkehrbar eindeutig (1:1 NAT), so entfällt dieser Schritt.

3.4 Common Gateway Interface (CGI)

Ein einfacher Webserver ist nur in der Lage, statische Seiten auszuliefern, die über einen URL angefordert werden. Das *Common Gateway Interface* ist ein Standard, um Daten zwischen dem Webserver und einer externen Anwendung auszutauschen. Der Webserver wird damit in die Lage versetzt, dynamische Informationen bereitzustellen, nämlich die Ausgabe eines CGI-Programms. Außerdem kann ein Web-Client auch Eingaben tätigen, die an ein CGI-Programm weitergereicht werden und dort bestimmte Aktionen auslösen.

Das CGI-Programm erhält seine Eingaben über Umgebungsvariablen oder die Standardeingabe, und es schickt seine Ausgaben über die Standardausgabe zurück an den Webserver und damit an den Client. [30]

3.5 Pluggable Authentication Modules (PAM)

Traditionell ist jede UNIX-Anwendung, die einen Dienst mit Benutzerauthentifizierung anbieten möchte, für die Implementierung dieser Authentifizierung selbst verantwortlich. Wenn Änderungen erforderlich sind, wie bei der Einführung eines neuen Authentifizierungsverfahrens oder beim Beheben von Sicherheitslücken in bestehenden Verfahren, so betreffen sie viele Anwendungen, und es sind dadurch Änderungen am Quelltext und eine Neukompilierung erforderlich.

Pluggable Authentication Modules (PAM) [31] ist ein Mechanismus, der die Benutzerauthentifizierung von den Anwendungsprogrammen trennt. Mit seiner Hilfe kann der Systemadministrator Authentifizierungsverfahren konfigurieren, ohne an den Anwendungen selbst etwas zu ändern. Die Anwendung lädt dann das gewünschte Modul zur Laufzeit, um die Authentifizierung durchzuführen. Weiterhin ermöglicht PAM, mehrere Authentifizierungsverfahren kombiniert zu verwenden, wobei vom Nutzer aber nur *eine* Authentifizierung gefordert wird (*unified login*).

3.6 Kerberos

Paßwortbasierte Authentifizierungsverfahren ermöglichen das Abhören und den Mißbrauch des Paßworts durch einen Angreifer, wenn keine besonderen Vorkehrungen getroffen wurden. Außerdem muß sich der Nutzer für jeden einzelnen Dienst neu authentifizieren, weswegen in der Vergangenheit sogar noch schwächere Verfahren angewandt wurden (rhosts, IDENT), die die Sicherheit weiter verringerten. Als Lösung empfiehlt sich der Einsatz von kryptographischen Verfahren, damit ein Angreifer keine Authentifizierung mehr fälschen kann.

Kerberos [32] ist ein Authentifizierungsverfahren, das diese Forderung erfüllt. Der Client authentifiziert sich gegenüber dem Anwendungsserver (Dienst) mit DES-verschlüsselten Nachrichten. Den dafür benötigten Sitzungsschlüssel erhalten beide Parteien von einem Authentifizierungsserver. Alle kritischen Daten werden dabei ebenfalls verschlüsselt übertragen. Somit ist nur noch eine Vertrauensstellung nötig: zwischen dem Authentifizierungsserver und den n Stationen, die sich authentifizieren möchten bzw. die eine Authentifizierung fordern. Die Komplexität wird dadurch von quadratisch $(n * (n-1)/2)$ auf linear (n) reduziert, und außerdem wird durch die Verschlüsselung überhaupt erst eine sichere Authentifizierung möglich.

Es wäre wünschenswert, daß sich ein Nutzer nur einmal authentifizieren muß und später während der Sitzung verschiedene Dienste nutzen kann, ohne daß eine erneute Paßworteingabe erforderlich ist (*single sign-on*). Zu diesem Zweck könnte auf dem Client das Paßwort des Nutzers nach der ersten Authentifizierung aufbewahrt werden und würde dann auch für alle nachfolgenden Authentifizierungsvorgänge zur Verfügung stehen. Diese Lösung ist aber gefährlich, deshalb werden bei Kerberos nur Sitzungsschlüssel mit begrenzter Gültigkeitsdauer gespeichert, um *single sign-on* zu realisieren. [33]

3.7 Dynamic Host Configuration Protocol

Der Anschluß eines Rechners ans Netzwerk erfordert die Konfiguration mehrerer Parameter, wie z. B. der IP-Adresse oder der Adresse des zuständigen Routers. Diese Konfiguration erfordert ein gewisses Maß an Fachwissen und verursacht bei einer größeren Anzahl von Rechnern auch einen beträchtlichen Aufwand. Während dieser bei fest angeschlossenen Rechnern noch vertretbar sein mag, so ist für nur zeitweise angeschlossene Rechner ein Verfahren wünschenswert, das diesen Vorgang automatisiert. Ein leistungsfähiges Verfahren ist das *Dynamic Host Configuration Protocol (DHCP)* [34], einen geringeren Funktionsumfang besitzen das *Reverse Address Resolution Protocol (RARP)* [35] und das *Bootstrap Protocol (BOOTP)* [36].

DHCP bietet folgende Funktionalität:

- Übermitteln statischer Konfigurationsinformationen (Subnetzmaske, Default Router, DNS-Server etc.)
- Verwalten und Übermitteln dynamischer IP-Adressen mit begrenzter Gültigkeitsdauer, automatisches Wiederverwenden der Adressen nach Ablauf ihrer Gültigkeit

DHCP-Client-Software ist für alle modernen Betriebssysteme verfügbar.

3.8 Domain Name System (DNS)

In der Anfangszeit des Internet wurde die Zuordnung zwischen Hostnamen und IP-Adressen zentral in einer Textdatei verwaltet, und diese dann per FTP auf alle Internet-Rechner kopiert. Der Aufwand dafür steigt quadratisch mit der Anzahl der Rechner, mit dem Wachstum des Internet wurde also eine besser skalierbare Lösung nötig. Zu diesem Zweck wurde das *Domain Name System (DNS)* entwickelt. [37] [38]

Das DNS definiert einen hierarchischen Namensraum und ordnet diesen Namen u. a. Netzwerkadressen (IPs) zu, die umgekehrte Zuordnung ist ebenfalls möglich. Die Verantwortlichkeit für einzelne Teilbäume der Hierarchie *(Zone)* kann *delegiert* werden, die Daten werden also nicht mehr zentral gespeichert, sondern verteilt auf den zuständigen DNS-Servern.

Wenn ein Anwendungsprogramm die Adresse zu einem Namen ermitteln möchte, ruft es eine Bibliotheksfunktion des *Resolver* auf. Der Resolver kennt den nächstgelegenen DNS-Server und leitet die Anfrage dorthin weiter. Ist dieser Server für die abgefragte Zone zuständig, beantwortet er die Anfrage selbst, ansonsten ermittelt er den zuständigen DNS-Server, der die Anfrage beantworten kann *(forward)*. Das Ergebnis erreicht umgekehrt auf dem gleichen Weg wieder das Anwendungsprogramm.

3.9 Internet Control Message Protocol (ICMP)

Das Internet-Protokoll (IP) dient zur Host-zu-Host-Kommunikation, ggf. über *Gateways* als Zwischenstation. Der Empfänger-Host und die Gateways können dem Absender eines IP-Datagramms Status- und Fehlermeldungen schicken, dazu dient das *Internet Control Message Protocol (ICMP)* [39].

Eine Einsatzmöglichkeit von ICMP ist die Erreichbarkeitsprüfung. Der Anfragende schickt dazu eine ICMP-Nachricht vom Typ *echo* (oft auch als *echo request* bezeichnet). Wenn der zu prüfende Rechner per IP erreichbar ist, antwortet er darauf mit einer Nachricht vom Typ *echo reply*.

3.10 Address Resolution Protocol (ARP)

IP-Datagramme werden oft über eine Verbindungsschicht auf Ethernet-Basis übertragen. Auch Wireless LANs nach IEEE 802.11 erscheinen für die höheren Schichten wie Ethernet. Im Normalfall (bei gerichteten Übertragungen, *unicast*) muß der Sender eines Ethernet-Frames die Ethernet-Adresse des Empfängers kennen. Weil eine statische Zuordnung von IP-Adressen zu Ethernet-(MAC-)Adressen nicht praktikabel wäre, wurde für diesen Zweck das *Address Resolution Protocol (ARP)* entwickelt. [40]

Ein Sender, die die MAC-Adresse eines Zielrechners ermitteln möchte, schickt eine ARP-Anfrage als Ethernet-Broadcast, welche die IP-Adresse des gesuchten Zielrechners enthält. Der Rechner mit dieser IP schickt eine ARP-Antwort zurück, aus der der ursprüngliche Absender die gewünschte MAC-Adresse erfährt.

In besonderen Fällen kann auch ein Rechner ARP-Anfragen beantworten, obwohl er nicht die nachgefragte IP-Adresse besitzt. Dieses Verfahren nennt man *Proxy ARP*. Seine Nutzung ist sinnvoll, wenn der antwortende Rechner als Router arbeitet, der das Paket an den eigentlichen Zielrechner weiterleiten kann, auf dem Quellrechner aber nicht die korrekte Route zum Zielrechner konfiguriert ist.

Eine weitere Anwendungsmöglichkeit für Proxy ARP ist das Unterdrükken von IP-Adressen, deren Nutzung im Subnetz unerwünscht ist. Ein Beispiel dafür könnten private Adreßbereiche [29] sein. Der Router leitet in diesem Fall die Daten auf sich selbst, um sie anschließend wegzuwerfen.

Kapitel 4

Praxis

Im folgenden wird die praktische Lösung der Aufgabe dargestellt. Zunächst wird die Software beschrieben, anschließend werden einige Entscheidungen erläutert, die bei der Implementierung getroffen wurden, und schließlich enthält diese Arbeit noch Hinweise zur Inbetriebnahme und Wartung des Systems.

4.1 Einordnung

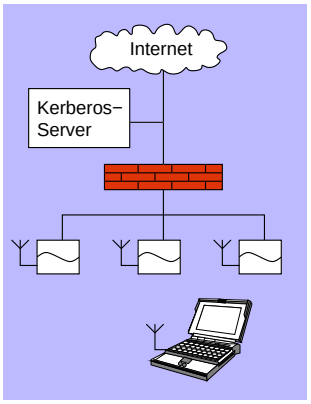

Abbildung 4.1: Funknetz mit Firewall

In Abbildung 4.1 ist dargestellt, wie sich die Lösung in die Netzstruktur einordnet. Alle Funktionen der Software sind auf einem einzigen Firewall-Rechner realisiert, der in der Grafik symbolisch als Ziegelmauer erscheint.

4.2 Komponenten

4.2.1 HTTP-Server

Auf dem Firewall-Rechner wird der HTTP-Server *Apache* eingesetzt, um das Anmeldeformular bereitzustellen und bei der Anmeldung ein CGI-Skript auszuführen, das alle nötigen Konfigurationsschritte vollzieht. Welche Schritte das im einzelnen sind, wird dann in Abschnitt 4.3.1 erläutert.

Der Nutzer authentifiziert sich gegenüber dem Server mit seinem URZ-Nutzerkennzeichen und -Paßwort. Diese Daten werden mit SSL/TLS verschlüsselt übertragen. Der Server kann die Authentifizierung nicht selbst prüfen, sondern verwendet das Kerberos-Protokoll, um die Authentifizierungsdaten vom AFS-Kerberos-Server des Rechenzentrums prüfen zu lassen. Auch diese Übertragung läuft verschlüsselt ab.

Damit das System möglichst einfach zu benutzen ist, kann der Nutzer eine beliebige Seite vom Webserver anfordern. Der Server ist so konfiguriert, daß er bei nicht vorhandenen Seiten keine Fehlermeldung ausgibt, sondern den Nutzer auf die Anmeldeseite umleitet. Der Vollständigkeit halber gilt das gleiche für Seiten, für die der Nutzer keine Berechtigung besitzt. Weiterhin ist das Anzeigen von Verzeichnisinhalten gesperrt, und der Server bietet überhaupt nur die zum Projekt gehörenden Seiten an. Somit ist sichergestellt, daß nicht durch Zufall eine andere als die Anmeldeseite abgerufen werden kann.

4.2.2 DHCP-Server

Bevor ein Client mit dem Webserver kommunizieren kann, benötigt er einen konfigurierten IP-Stack. Besonders einfach ist das mit DHCP realisierbar, weshalb dieses Protokoll auch hier zum Einsatz kommt. Ein neuer Client bekommt vom DHCP-Server eine dynamische IP-Adresse aus dem privaten Adreßbereich [29] zugewiesen, außerdem werden Netzmaske, Broadcast-Adresse, *default router* und Nameserver übermittelt. Als Router und Nameserver gibt der DHCP-Server dabei die eigene IP-Adresse (des Firewall-Rechners) an. Die Daten werden mit einer kurzen Gültigkeitsdauer versehen (momentan 1 Minute), damit Änderungen kurzfristig beim Client wirksam werden können. Solche Änderungen sind zum Beispiel nötig, wenn sich der Client dafür entscheidet, im Zuge der Anmeldung auf eine offizielle IP-Adresse umzuschalten.

4.2.3 DNS-Server

Eine weitere Voraussetzung zur Kommunikation mit dem Webserver und zur sinnvollen Nutzung des Netzzugangs ist eine funktionierende Namensauflösung (DNS). Dazu werden hier zwei Funktionen realisiert: der Zugriff

auf die weltweit gültigen DNS-Informationen, sowie die Zuordnung von Namen zu den privaten, dynamisch vergebenen IP-Adressen.

Für den Zugriff auf weltweite DNS-Informationen arbeitet der DNS-Server als Proxy, der Client stellt also alle Anfragen an den lokalen DNS-Server auf dem Firewall-Rechner, und dieser leitet sie weiter. Die Proxy-Funktionalität wird benötigt, weil ein Client vor seiner Anmeldung noch keine Daten nach außerhalb des Funknetzes übertragen und somit auch noch keine anderen DNS-Server erreichen kann. Das Anmeldeformular soll aber durch den Abruf einer Webseite von einem beliebigen Server angezeigt werden. Dabei versucht der Client, die IP-Adresse dieses externen Servers zu ermitteln. Letztlich wird die HTTP-Anfrage zwar nicht an diese IP geschickt, sondern an den WWW-Server auf dem Firewall-Rechner umgelenkt, trotzdem ist aber ein uneingeschränkter Zugriff auf das DNS zur Anmeldung erforderlich.

Für den privaten Bereich des Namensraumes ist der Server selbst zuständig, und die entsprechenden Informationen sind nur innerhalb des Funknetzes abrufbar. Den privaten IP-Adressen werden dabei nach einem festen Schema Namen zugeordnet, es findet also kein Abgleich zwischen DNS-Server und DHCP-Server statt (dynamisches DNS). Es wäre auch nicht besonders sinnvoll zu versuchen, eine feste Zuordnung zwischen Rechner und Name herzustellen, denn das System wurde ja gerade für den Netzzugang von zunächst unbekannten Rechnern entwickelt.

4.2.4 Skripte

Alle bisher genannten Komponenten sind Standardsoftware, die wie benötigt konfiguriert wurde. Die beschriebene Lösung enthält außerdem noch selbstentwickelte Software, im wesentlichen besteht diese aus Skripten und Firewallregeln. Die Skripte werden in den Abschnitten 4.3 und 4.5.3 (Teilabschnitt *Ausführbare Dateien*) näher beschrieben. Sie umfassen folgende Funktionen:

- Initialisierung nach Installation des Softwarepakets
- Systemstart/Herunterfahren
- WWW-Interface zur Anmeldung: Formular und CGI-Skript
- automatische Abmeldung
- automatischer Neustart des DHCP-Servers nach Änderung seiner Konfiguration
- Kommoandozeileninterface für Administratoren

4.2.5 Firewall-Regeln

Die grundlegende Funktion der Firewall-Regeln besteht darin, nur angemeldeten Rechnern Zugriff auf das Internet zu gewähren. Anfragen nicht angemeldeter Rechner werden größtenteils gesperrt, teilweise aber auch auf den Firewall-Rechner umgeleitet.

Die Umleitung umfaßt DNS-Anfragen sowie Anfragen an die Standardports für WWW (HTTP, HTTPS) und für WWW-Proxies (8080). Sie ermöglicht, daß der Nutzer eine beliebige WWW-Seite abrufen kann, zum Beispiel seine Browser-Startseite. Im Zusammenspiel mit der bereits erläuterten Webserver-Konfiguration wird daraufhin immer die Anmeldeseite geliefert.

Welche Zugriffsrechte ein angemeldeter Rechner erhält, hängt von der Nutzerklasse ab, die bei der Anmeldung gewählt wurde. Während für Nutzer der Klassen *public* und *private* keine Beschränkungen konfiguriert sind, hat ein Gast nur eingeschränkte Rechte. Zur Zeit dürfen Gäste nur WWW-Seiten von Servern innerhalb der TU Chemnitz abrufen. Die Klassifizierung eines Rechners erfolgt anhand der MAC- und IP-Adresse.

4.3 Funktionsweise

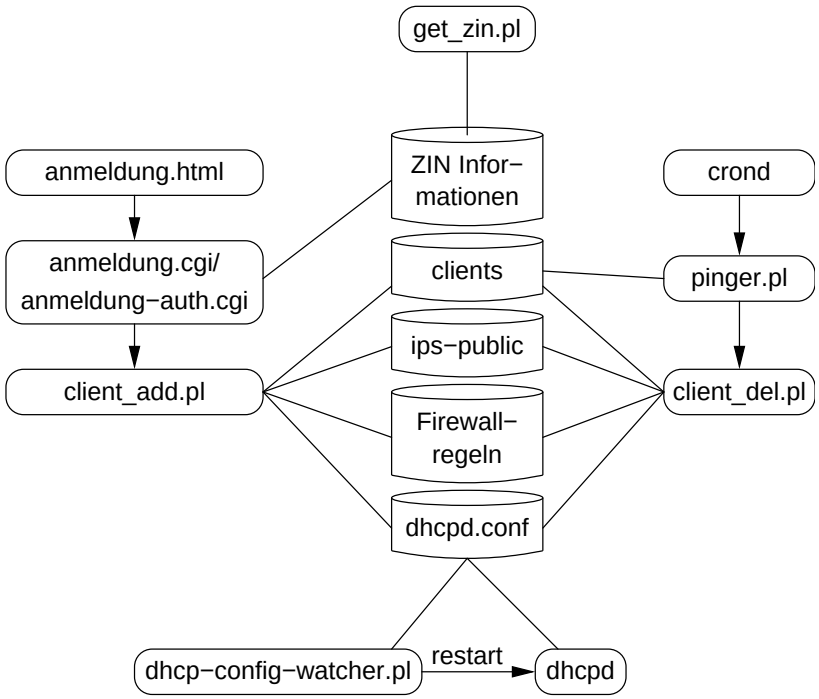

Abbildung 4.2: Struktur der Firewall-Software

Abbildung 4.2 zeigt, wie die wesentlichen Komponenten der Software zusammenarbeiten. Der linke Teil der Grafik stellt die Anmeldung dar, der rechte Teil die automatische Abmeldung. In der Mitte sind die Zustandsspeicher dargestellt, mit Ausnahme der Firewall-Regeln handelt es sich dabei um Text-Dateien. Der untere Teil zeigt, wie Änderungen an der DHCP-Konfiguration aktiviert werden, und der obere Teil steht für das Kopieren der ZIN-Informationen aus dem URZ. Im folgenden sind nur diejenigen Komponenten dargestellt, die für den unbeaufsichtigten Betrieb der Software erforderlich sind. Eine Anleitung für Administratoren findet sich dagegen in Abschnitt 4.5.

4.3.1 Registrierung eines Clienten

Ein neuer Client erhält zunächst seine IP-Konfiguration vom DHCP-Server (siehe Abschnitt 4.2.2). Wenn der Nutzer anschließend eine beliebige WWW-Seite aufruft, z. B. die Startseite seines Browsers, werden 2 Anfragen generiert: eine DNS- und anschließend eine HTTP-Anfrage.

Zunächst ermittelt der Browser aus dem URL den HTTP-Server (meist als Hostname), dann fragt er den DNS-Server nach der IP-Adresse des HTTP-Servers. Im Normalfall steht dieser Webserver außerhalb des Funknetzes, der DNS-Server kennt deshalb die Antwort nicht selbst, sondern arbeitet als Proxy (siehe Abschnitt 4.2.3). Falls der Client einen DNS-Server außerhalb des Funknetzes fest konfiguriert hat, wird die DNS-Anfrage durch die Firewall-Regeln auf den lokalen DNS-Server umgeleitet, um auch in diesem Fall eine Anmeldung zu ermöglichen.

Der Browser kennt jetzt die IP-Adresse des Zielservers und baut eine HTTP-Verbindung dorthin auf. Diese Verbindung wird ebenfalls mit Hilfe von Firewall-Regeln auf den lokalen Webserver umgeleitet (siehe Abschnitt 4.2.5). Die HTTP-Anfrage enthält jetzt noch einen Pfad zur angeforderten Datei, diese wird aber höchstwahrscheinlich auf dem lokalen Webserver nicht existieren, also liefert der Webserver eine Fehlerseite. Die Apache-Option `ErrorDocument` mit dem kompletten URL der Anmeldeseite bewirkt schließlich, daß der Browser dorthin umgeleitet wird und die Anmeldeseite darstellt. Zur Zeit ist die Umleitung so konfiguriert, daß bereits die Anmeldeseite per HTTPS ausgeliefert wird, das ist aber genaugenommen erst bei der folgenden Seite nötig.

Der Nutzer wählt im Anmeldeformular seine Klasse und füllt nach Bedarf weitere Felder aus. Je nach Nutzerklasse werden die Daten dann an das entsprechende CGI-Skript geschickt: `anmeldung.cgi` bei der Anmeldung als Gast, `anmeldung-auth.cgi` bei den übrigen Nutzerklassen. Der HTTP-Server ist so konfiguriert, daß das Skript `anmeldung-auth.cgi` nur mit Authentifizierung abgerufen werden kann. Der Ablauf der Authentifizierung selbst wurde in Abschnitt 4.2.1 erläutert.

Tatsächlich handelt es sich bei den beiden Skripten um die gleiche Datei

(im folgenden als „das CGI-Skript“ bezeichnet), die in Abhängigkeit von der Nutzerklasse verschiedene Funktionalität zur Verfügung stellt. Um eine daraus resultierende Schwachstelle zu vermeiden, prüft das CGI-Skript für die privilegierten Nutzerklassen selbst noch einmal, ob die Authentifizierung tatsächlich erfolgt ist.

Das CGI-Skript

Zu Beginn prüft das Skript die Nutzerklasse. Eine Anmeldung als Gast ist nur möglich, wenn das in der Konfigurationsdatei erlaubt wurde und der Nutzer die zusätzlichen Felder ausgefüllt hat (Vorname, Name, Zweck). Für eine Anmeldung mit einer anderen Nutzerklasse wird dagegen überprüft, ob sich der Nutzer authentifiziert hat und ob er ein Zertifikat Internet-Nutzung (ZIN) besitzt.

Anschließend ermittelt das Skript die MAC-Adresse des Clienten, indem es dessen IP-Adresse im lokalen ARP-Cache (`/proc/net/arp`) sucht, und es prüft die CGI-Parameter auf ungültige Zeichen. Wenn bis jetzt alle Schritte erfolgreich waren, ruft es `client_add.pl` mit Root-Rechten (per `sudo`) auf und übergibt MAC-Adresse, Nutzerklasse und die momentan benutzte IP des Clienten als Parameter. Am Rückgabewert von `client_add.pl` erkennt das CGI-Skript, ob die Registrierung erfolgreich war. Es schreibt einen entsprechenden Eintrag ins Logfile und gibt eine WWW-Seite mit dem Status der Anmeldung aus.

Skript `client_add.pl`

Dieses Skript wird normalerweise vom CGI-Skript aus aufgerufen (siehe voriger Teilabschnitt), es kann aber auch vom Administrator (root) direkt auf der Kommandozeile ausgeführt werden. Falls die Nutzerklasse *private* oder *public* ist, entnimmt es eine verfügbare offizielle IP-Adresse aus der Datei `ips-public`. In jedem Fall wertet es seine Kommandozeilenparameter aus und trägt die Informationen in eine neue Zeile der Datei `clients` ein. Wenn der Client im Funknetz mit einer offiziellen IP-Adresse arbeiten soll (also ohne NAT), dann wird noch die DHCP-Konfigurationsdatei `dhcpd.conf` aktualisiert. `client_add.pl` liest anschließend die Firewall-Dateien im Verzeichnis `/etc/wlan-firewall/client`, ersetzt die Variablen darin durch ihre aktuellen Werte (MAC, IP, Nutzerklasse) und fügt die resultierenden Firewall-Regeln per `iptables-restore` ins laufende System ein. Welche Firewall-Dateien gelesen werden und in welcher Reihenfolge, legt die Konfigurationsvariable `%templates` in `/etc/wlan-firewall/FW_Config.pm` fest.

Wenn eine dieser Aktionen fehlschlägt, werden alle bisher unternommenen Schritte rückgängig gemacht, um einen konsistenten Zustand zu bewahren. Diese Vorgehensweise ist mit dem Abbruch einer Datenbank-Transaktion (*rollback*) vergleichbar. `client_add.pl` beendet sich in diesem Fall

mit einem Rückgabewert größer Null und gibt eine Fehlermeldung aus.

Falls alle Aktionen erfolgreich waren, ist der Rückgabewert Null, und es werden die Client-Daten ausgegeben, im gleichen Format wie in der Datei `clients`. Auf diesem Weg erfährt auch das CGI-Skript die neue IP-Adresse des Clienten, wenn sie sich durch die Anmeldung geändert hat.

DHCP-Aktualisierung

Die eingesetzte DHCP-Serversoftware (ISC DHCP 2.0) ist nicht in der Lage, Änderungen an ihrer Konfiguration im laufenden Betrieb zu übernehmen — weder automatisch noch nach Aufforderung durch Senden eines Signals. Aus diesem Grund ist nach jeder Änderung ein Neustart des DHCP-Servers erforderlich. Um die CPU-Last auf dem Firewall-Rechner zu begrenzen und eine hohe Verfügbarkeit des DHCP-Dienstes zu gewährleisten, soll zwischen zwei Neustarts eine gewisse Mindestpause liegen, zur Zeit 10 Sekunden. Deshalb wird der DHCP-Server nicht direkt nach jeder Änderung neu gestartet, sondern der Daemon-Prozeß `dhcp-config-watcher.pl` überwacht die Konfigurationsänderungen und sorgt bei Bedarf für den Neustart und die Einhaltung der Pausen.

4.3.2 Erreichbarkeitsprüfung und Deregistrierung

Theoretisch ist der Mißbrauch eines freigeschalteten Zugangs durch Außenstehende möglich, weil keine starke Authentifizierung der Daten erfolgt. Weiterhin steht nur ein relativ knapper Vorrat an offiziellen IP-Adressen zur Verfügung. Deshalb soll eine Anmeldung nur befristet gültig sein, davon ausgenommen sind lediglich Infrastruktur-Geräte wie Access Points, die von einem Administrator registriert wurden.

Ein sinnvolles Kriterium zur Verlängerung einer Freischaltung könnte sein, daß der Client den Netzzugang noch aktiv nutzt. Um aber auch Pausen zu ermöglichen, in denen er zwar Verbindung zum Netz besitzt, aber keine Daten überträgt, soll stattdessen regelmäßig seine Erreichbarkeit mit `ping` geprüft werden.

Skript `pinger.pl`

Jede Minute wird dieses Skript per Cron-Job gestartet. Zuerst ermittelt es aus der Datei `clients` die IP-Adressen der Clienten, die auf Erreichbarkeit geprüft werden müssen. Dabei werden die Spalten für die Nutzerklasse und für die letzte Änderung (`last_change`) berücksichtigt. Für jeden der gefundenen Clienten startet es dann ein `ping`-Kommando als Subprozeß. Der Start der einzelnen `ping`-Subprozesse erfolgt zeitversetzt, um die Last auf dem Firewall-Rechner und die Netzlast gering zu halten. Die Verzögerung wird dabei so gewählt, daß die Bearbeitung aller ermittelten IP-Adressen trotzdem in weniger als einer Minute abgeschlossen ist.

Für alle Clienten, die die Ping-Anfrage nicht beantwortet haben, wird ein client-spezifischer Zähler in `clients` erhöht; für alle Clienten, die geantwortet haben, wird dieser Zähler auf Null zurückgesetzt. Wenn der Zähler für einen Clienten seinen Maximalwert (`$ping_retries`, z. Zt. 5) erreicht, wird `client_del.pl` mit der IP-Adresse dieses Clienten als Parameter aufgerufen, um ihn zu deregistrieren.

Skript `client_del.pl`

Dieses Skript hebt eine Registrierung auf und ist damit das Gegenstück zu `client_add.pl`. Es erwartet eine IP-Adresse als Kommandozeilenparameter, sucht den zugehörigen Client in `clients` und löscht ihn dort. Es gibt die gefundenen Daten aus und schreibt diese außerdem zusammen mit einer Statusmeldung in die Logdatei. Anschließend schreibt es die offizielle IP-Adresse zurück in die Datei `ips-public`, falls bei der Anmeldung eine IP entnommen wurde. Diese Datei wird als Warteschlange (FIFO) verwaltet, damit die soeben freigewordene IP-Adresse als letzte wieder neu vergeben wird. In der Nutzerklasse `public` läßt sich mit diesem Verfahren mit hoher Wahrscheinlichkeit verhindern, daß eine IP-Adresse neu verwendet wird, bevor die Gültigkeitsdauer der zugehörigen DHCP-Lease abgelaufen ist.

Analog zu `client_add.pl` aktualisiert das Skript dann bei Bedarf die Datei `dhcpd.conf`, generiert Kommandos zum Löschen der Firewall-Regeln und führt diese per `iptables-restore` aus. Den Erfolg dieser Aktionen zeigt es per Exit-Code an, außerdem gibt es eine Statusmeldung aus. Im Fehlerfall ist hier kein Rollback vorgesehen — das Skript gibt nur eine Warnung aus und setzt seine Arbeit fort, zum Schluß wirkt sich der Fehler aber auf den Exit-Code aus.

Wenn sich die DHCP-Konfiguration geändert hat, bemerkt dies der Daemon-Prozeß `dhcp-config-watcher.pl` und startet den DHCP-Server neu, wie bereits beschrieben.

4.3.3 Firewall-Regeln und Nutzerklassen

Im Rahmen dieser Arbeit ist die Unterscheidung in Nutzerklassen mit verschiedenen Rechten gefordert. Die Klassen zeichnen sich durch folgende Merkmale aus:

- `guest`: Es wird Masquerading (n:1 NAT) eingesetzt. Damit sind alle Clienten dieser Klasse von außen unter der externen IP-Adresse des Firewall-Rechners sichtbar. Ein Verbindungsaufbau von außen zum Client ist nicht möglich. Zur Zeit ist nur der WWW-Zugriff innerhalb der TU-Chemnitz gestattet.

- `private`: Hier kommt 1:1 NAT zum Einsatz, um eine Änderung der IP-Adresse bei der Anmeldung zu vermeiden. Der Client kann seine

private IP-Adresse behalten und ist von außen unter einer eigenen offiziellen IP erreichbar. Ein Verbindungsaufbau ist in beide Richtungen möglich. Protokolle wie FTP, die normalerweise nicht mit NAT funktionieren, werden teilweise über Kernelmodule unterstützt.

- `public`: Die Auswahl dieser Nutzerklasse ist sinnvoll, wenn NAT nicht erwünscht ist. Nach der Anmeldung muß der Nutzer darauf warten, daß die Gültigkeitsdauer seiner privaten IP-Adresse abläuft und sein DHCP-Client die IP-Adresse aktualisiert. Alternativ kann er auch die Aktualisierung manuell auslösen.

- `static`: Für *Access Points* und ähnliche Geräte im Subnetz, die zu Management-Zwecken immer erreichbar sein sollen, wurde schließlich noch diese Nutzerklasse eingeführt. Die Erreichbarkeit wird nicht geprüft, es erfolgt also auch keine automatische Deregistrierung. Datenübertragungen sind nur zu Rechnern innerhalb der TU möglich.

Die Firewall-Regeln sind mit *netfilter* im Linux-Kernel 2.4 realisiert. Sie benötigen nur einen Teil der verfügbaren *chains*; in Abbildung 3.1 finden sich diese im unteren Bereich. Jede der genutzten Chains wird gleich im Detail besprochen, weiterhin wird zwischen hereinkommenden und ausgehenden Paketen unterschieden. Je nach Richtung betreten bzw. verlassen diese Pakete den Firewall-Rechner über andere Netzwerk-Interfaces; sie durchlaufen die Firewall-Chains aber trotzdem immer in der gleichen Reihenfolge wie in der Grafik angegeben. Als *ausgehend* sollen Pakete bezeichnet werden, die von einem Rechner im Funknetz erzeugt wurden und den Firewall in Richtung Campusnetz/Internet passieren. Die umgekehrte Richtung soll *hereinkommend* heißen.

- `mangle/PREROUTING`: Die Regeln in dieser Chain klassifizieren vom Netz empfangene Pakete, indem sie sie mit einer numerischen Markierung (*firewall mark*) versehen, die der Nutzerklasse entspricht. Diese Markierung wird dann in den übrigen Chains wieder ausgewertet. Bei ausgehenden Paketen werden zur Einordnung Quell-MAC und Quell-IP herangezogen, bei hereinkommenden Paketen dagegen nur die Ziel-IP, weil sie den Rechner über das externe Interface betreten und die MAC-Adresse des Clienten dort keine Rolle spielt.

  Hereinkommende Pakete der Nutzerklassen `private`, `public` und `static` besitzen für jeden Client eine eindeutige Ziel-IP. Für jede dieser IP-Adressen wird bei der Anmeldung eine eigene Firewall-Regel eingerichtet, um die zugehörige Markierung zu setzen. Pakete, die die externe IP-Adresse des Firewall-Rechners als Ziel tragen, werden pauschal der Nutzerklasse `guest` zugeordnet. Das ist wichtig, wenn später einmal weitere Gast-Nutzerklassen hinzugefügt werden sollen. Eine Un-

terscheidung mehrerer Gast-Klassen ist mit der vorliegenden Software nur für ausgehende Pakete möglich.

- `nat/PREROUTING`: Die `PREROUTING`-Chain in der Tabelle `nat` ist für Destination NAT (DNAT) zuständig. Die Regeln für ausgehende Pakete sorgen für die Umleitung einiger Protokolle (HTTP, HTTPS, http-proxy, DNS) auf den Firewall-Rechner selbst, falls die Anfrage von einem unbekannten Client stammt. Bei hereinkommenden Paketen wird die IP-Zieladresse ersetzt (1:1 NAT), wenn sie zu einem Client der Nutzerklasse `private` gehört. Das betrifft nur Verbindungen, die von außen aufgebaut werden, die andere Richtung wird in `POSTROUTING` behandelt (siehe dort).

- `filter/FORWARD`: Hier werden die Rechte der einzelnen Nutzerklassen festgelegt. Zur Zeit sind nur Einschränkungen für ausgehende Pakete definiert, und auch nur für die Nutzerklassen `guest` und `static`. Ein Gast darf nur auf WWW-Server der TU Chemnitz zugreifen, die auf den Standard-Ports (HTTP, HTTPS) erreichbar sind. In der Klasse `static` sind nur Übertragungen zu Rechnern innerhalb der Uni erlaubt.

- `nat/POSTROUTING`: Diese Chain ist für Source NAT (SNAT) zuständig und kommt hier nur zum Einsatz bei Verbindungen, die von innen aufgebaut werden. Bei Paketen der Nutzerklasse `guest` wird die IP-Quelladresse durch die externe IP des Firewall-Rechners ersetzt. Für die Klasse `private` wird dagegen 1:1 SNAT realisiert, damit jeder Client dieser Klasse nach außen unter einer eigenen offiziellen IP-Adresse sichtbar ist, während er intern seine private IP-Adresse weiternutzt.

4.3.4 ZIN-Informationen

Ob ein Nutzer das Zertifikat Internet-Nutzung (ZIN) besitzt oder davon befreit ist, ist in der URZ-Datenbank gespeichert. Daraus generiert das URZ regelmäßig Textdateien und hinterlegt diese im Dateisystem (AFS). Der Cron-Job `get_zin.pl` auf dem Firewall-Rechner holt stündlich diese Dateien, faßt sie zu einer zusammen und legt sie lokal ab. Das CGI-Skript für die Anmeldung eines Nutzers überprüft dann, ob sein Nutzerkennzeichen in dieser Datei enthalten ist und er damit die Berechtigung besitzt, das Funknetz zu nutzen. Bei einer Anmeldung als Gast entfällt diese Prüfung, denn diese Klasse ist hauptsächlich für Nutzer vorgesehen, die keinen URZ-Zugang besitzen.

4.4 Entwurfsentscheidungen

4.4.1 Ähnliche Software als Grundlage

Die Ethernet-Dosen in öffentlichen Bereichen der TU Chemnitz werden zur Zeit mit der Portmanager-Software [41] von André Breiler verwaltet. Diese ändert die Zuordnung von Dosen zu VLANs, um Clienten bestimmte Rechte zu gewähren. In einem drahtlosen Netz ist ein Client aber nicht mehr an eine bestimmte Dose angeschlossen, und die eingesetzten Access Points können selbst keine VLANs verwalten. Die vorliegende Arbeit nutzt deshalb Paketfilter-Regeln, um eine ähnliche Funktionalität zu realisieren. Weitere Ideen wie die Nutzerklassen oder die Erreichbarkeitsprüfung wurden ebenfalls aus [41] übernommen, aber neu implementiert.

In der Portmanager-Software waren Probleme mit der NAT-Implementierung von Linux aufgetreten, sie wurden dort mit Hilfe eines speziellen Daemons (`udp_spoofer`) umgangen, indem dieser Pakete mit falscher Absenderadresse erkannte und ein zweites Mal in korrigierter Form schickte. Bei ersten Tests für die vorliegende Firewall-Software ließen sich die NAT-Probleme nicht mehr nachvollziehen. Vermutlich ist der Fehler in aktuelleren Kernel-Versionen behoben. Der Einsatz des `udp_spoofer` ist also nicht nötig.

Weiterhin realisiert die Portmanager-Software eine Proxy-ARP-Funktion, um jegliche Datenübertragung im Subnetz auf den Anmelderechner umzuleiten. Das ist dort nötig, um die Anmeldung von Rechnern der Klasse `URZ` zu ermöglichen, weil diese bereits vor der Anmeldung mit einer IP-Adresse aus einem beliebigen Subnetz der TU Chemnitz arbeiten. Wegen der fehlenden VLAN-Funktionalität konnte diese Klasse hier nicht realisiert werden, es ist also kein Proxy ARP mehr nötig. Die beiden getrennten IP-Bereiche mit privaten und offiziellen Adressen im Funknetz benötigen ebenfalls kein Proxy ARP, sie lassen sich mit einer zweiten IP-Adresse auf dem internen Netzwerkinterface des Firewall-Rechners realisieren (als Alias-Interface konfiguriert).

Proxy ARP könnte zwar immer noch als rudimentärer Sicherheitsmechanismus eingesetzt werden, um eine Kommunikation zwischen nicht angemeldeten Rechnern im Funknetz zu verhindern. Gegebenenfalls antworten dann allerdings 2 Rechner einander widersprechend auf ARP-Anfragen. Außerdem erschwert Proxy ARP die Fehlersuche, weil es das System komplexer macht. Aus diesen Gründen kommt es hier nicht zum Einsatz. Eine bessere Möglichkeit, um nicht angemeldeten Rechnern die Kommunikation innerhalb des Funknetzes zu verwehren, wird im Abschnitt 4.4.7 beschrieben.

4.4.2 Standardsoftware

NAT

Eine 1:1 Network Address Translation kann man auf verschiedenen Wegen erreichen: zustandslos mit *policy routing* oder zustandsbehaftet mit Netfilter und *connection tracking*. Policy Routing ändert nur die IP-Adressen der Pakete, aber nicht die Portnummern. Da es keine Zustandsinformationen speichern oder auswerten muß, ist es schnell und ressourcenschonend, aus dem gleichen Grund ist damit aber keine Sonderbehandlung für NAT-untaugliche Protokolle möglich.

Für Connection Tracking gilt das Gegenteil: mit Hilfe von Zusatzmodulen können problematische Protokolle wie FTP mit NAT in Einklang gebracht werden, und es verbraucht mehr Ressourcen. Bei Bedarf ändert Netfilter auch Portnummern, denn es ist nicht nur für 1:1 NAT, sondern auch für n:1 NAT einsetzbar.

Da die Software sowieso n:1 NAT und damit Connection Tracking benötigt, bringt es vermutlich kaum Vorteile, für 1:1 NAT Policy Routing einzusetzen. Deshalb kommt Netfilter auch für 1:1 NAT zum Einsatz.

DHCP

Es gibt im wesentlichen zwei DHCP-Server-Implementierungen für Linux: vom Internet Software Consortium (ISC) [42] und von der Carnegie Mellon University (CMU) [43]. Die CMU-Software beinhaltet einen BOOTP-Server, der um DHCP ergänzt wurde. Sie wird offenbar nicht mehr weiterentwickelt.

Die DHCP-Software des ISC ist vor kurzem in der Version 3.0 erschienen, in dieser Arbeit kommt noch der Server aus Version 2.0 zum Einsatz. Sowohl der DHCP-Server als auch der DHCP-Client besitzen einen größeren Funktionsumfang als andere Implementierungen. Bevor der Server eine dynamische IP-Adresse vergibt, prüft er mit `ping`, ob sie bereits benutzt wird. Im Notfall kann deshalb ein Client auch mit einer freien statischen IP-Adresse (ohne Nutzung von DHCP) arbeiten, ohne die dynamische Adreßvergabe zu stören. Außerdem versucht der Server, einem Client immer wieder die gleiche IP-Adresse zuzuweisen, solange diese noch frei ist. Zu diesem Zweck merkt sich der Server die IP-Zuordnungen auch über den Ablauf ihrer Gültigkeit hinaus.

Ein Test mehrerer DHCP-Client-Implementierungen (Windows, pump, dhcpcd, ISC) fiel ebenfalls zugunsten ISC aus, deshalb sei an dieser Stelle auch den Nutzern des Funknetzes die ISC-Software empfohlen. Details zu diesem Test finden sich im Anhang.

4.4.3 Programmiersprache

Aktuelle Programmiersprachen bieten im Normallfall eine ausreichende Funktionalität, so daß diese kein Entscheidungskriterium bei der Wahl der Sprache sein muß. Die Performance ist beim vorliegenden Projekt zwar wichtig, die kritischen Stellen lassen sich aber durch die Wahl einer anderen Sprache kaum optimieren (mehr dazu in Abschnitt 4.4.9).

Als ein wesentliches Kriterium für die Auswahl der Programmiersprache kann dagegen die Abstraktionsebene der Sprache dienen, d. h. ob sich komplexe Sachverhalte darin einfach ausdrücken lassen. Deshalb ist eine Skriptsprache wie Perl oder Python einer traditionellen Sprache wie C vorzuziehen. Sie bietet außerdem den Vorteil kürzerer Testzyklen, weil Änderungen keine Neukompilierung erfordern.

Die Wahl fiel bei diesem Projekt auf Perl, weil es sowohl dem Autor als auch den verantwortlichen Mitarbeitern des Rechenzentrums vertraut ist. Rückblickend läßt sich allerdings feststellen, daß Perl trotz des überschaubaren Projektumfangs hier bereits zu Einschränkungen führt. Formale Funktionsparameter, Exceptions und Objektorientierung lassen sich damit nur umständlich realisieren.

4.4.4 Authentifizierung

Die Anmeldung ist mit einem WWW-Formular realisiert, damit sie plattformunabhängig nutzbar ist und keine zusätzliche Software auf dem Nutzerrechner erfordert. Das gleiche soll für die Authentifizierung gelten, deshalb kommt hier ebenfalls ein HTTP-Mechanismus zum Einsatz. Zur Auswahl stehen die Verfahren *basic* und *digest* [44]. Die Digest-Authentifizierung wird allerdings noch nicht von allen Browsern unterstützt, außerdem sind dafür auf dem HTTP-Server MD5-Paßwort-Hashes erforderlich, die sich nicht aus der URZ-Datenbank gewinnen lassen. Damit würde ein unvertretbar hoher Aufwand zur Verwaltung der Authentifizierungsdaten erforderlich.

Die Basic-Authentifizierung überträgt das Paßwort im Klartext. Der Server kann somit ein beliebiges Verfahren einsetzen, um das Paßwort zu prüfen. Aus dem gleichen Grund ist aber auch eine verschlüsselte Übertragung nötig, um das Paßwort gegen Abhören zu sichern. Deshalb kommt hier die Basic-Authentifizierung über HTTP zum Einsatz, wobei die Verbindung mit SSL/TLS geschützt wird.

Im einfachsten Fall würde der Server dann eine Datei mit den verschlüsselten UNIX-Paßworten verwenden, um das Nutzerpaßwort zu prüfen. Dieses Verfahren hat aber den Nachteil, daß nur die ersten 8 Zeichen des Paßwortes signifikant sind.

Der HTTP-Server nutzt deshalb Kerberos, um das Paßwort zu prüfen. Er tritt dabei im Kerberos-Protokoll als Client auf. Die HTTP-Serversoftware Apache unterstützt selbst keine Kerberos-Authentifizierung, deshalb wickelt

Apache die Authentifizierung über PAM ab [45]. Für PAM wiederum existiert ein Modul zur Authentifizierung gegenüber dem AFS-Kerberos-Server [46].

4.4.5 Firewall-Regeln

firewall mark

Ursprünglich war vorgesehen, die Einordnung in die Nutzerklasse dort vorzunehmen, wo eine Firewall-Entscheidung zu treffen ist, also zum Beispiel in der `FORWARD`-Chain. Es stellte sich allerdings heraus, daß die Prüfung der MAC-Adresse nur in `mangle/PREROUTING` zuverlässig funktioniert, deshalb ist hier der Einsatz von *firewall marks* nötig, um das Ergebnis der Prüfung an andere Chains weitergeben zu können. Rückblickend betrachtet erscheint dem Autor die Nutzung von *firewall marks* auch deswegen sinnvoll, weil damit die Einordnung zentral erfolgt und das Ergebnis mehrfach genutzt werden kann.

Falsche IP- oder MAC-Adressen

Ein Paket von einem registrierten Rechner wird nur dann in die entsprechende Nutzerklasse eingeordnet, wenn sowohl die MAC- als auch die IP-Adresse mit der Registrierung übereinstimmt, ansonsten gilt der Client als nicht angemeldet. Wenn weder die MAC- noch die IP-Adresse registriert sind, ist dieses Verhalten auch einleuchtend. Was geschieht aber in den übrigen beiden Fällen?

Eine falsche MAC-Adresse in Verbindung mit einer registrierten IP kann auf einen Mißbrauch oder die versehentliche Nutzung einer fremden IP-Adresse hindeuten. Dieser Fall wird momentan ignoriert. Der Angreifer kann zwar die Kommunikation des rechtmäßigen IP-Nutzers stören, aber selbst keine Ressourcen nutzen. Als Erweiterungsmöglichkeit wäre denkbar, Informationen über die Pakete des Angreifers aufzuzeichnen und später auszuwerten.

Eine falsche IP-Adresse und eine registrierte MAC können verschiedene Ursachen haben. Bei einer Anmeldung für die Nutzerklasse `public` ändert sich die Client-IP, es ist aber nicht garantiert, daß diese Änderung in den Firewall-Regeln und auf dem Nutzerrechner synchron vollzogen wird. Eine weitere Ursache kann darin liegen, daß jemand mehrere per Kabel angeschlossene Rechner über einen Router mit Wireless LAN-Interface anmelden möchte. Diese Konfiguration wird nicht unterstützt, weil die Software pro MAC-Adresse nur eine Client-Anmeldung zuläßt; statt des Routers muß der Nutzer eine Bridge einsetzen oder auf dem Router Masqerading (n:1 NAT) aktivieren. Eine dritte Möglichkeit wäre noch ein Mißbrauch der fremden MAC-Adresse oder die versehentliche Auslieferung von Netzwerkkarten mit gleicher MAC-Adresse durch den Hersteller. Die letztgenannten Ursachen

dürften relativ selten auftreten, und die ersten beiden Ursachen sind kein Grund zur Besorgnis. Deshalb werden auch im Fall einer falschen IP-Adresse keine Maßnahmen ergriffen.

4.4.6 Funktionsweise

Prüfung auf bereits registrierte IPs

Der DHCP-Server stellt sicher, daß eine IP-Adresse nicht gleichzeitig mehr als einmal vergeben wird. Das schützt aber nur dann vor doppelten IP-Adressen, wenn alle Clienten DHCP verwenden. Zur Sicherheit wird deshalb zusätzlich bei der Anmeldung geprüft, ob die gleiche IP-Adresse schon registriert ist. Diesen Test könnte man noch ausweiten, zum Beispiel wäre es denkbar, nicht nur registrierte IP-Adressen, sondern auch schon die vom DHCP-Server vergebenen IP-Adressen (*leases*) mit einzubeziehen. Im Normalfall sollte der Test aber vollkommen ausreichen.

Ermittlung der MAC-Adresse

Beim Anmeldevorgang gibt es 2 Stellen, an denen die MAC-Adresse eines Clienten sinnvoll ermittelt werden kann: im CGI-Skript oder in `client_add.pl`. Würde die MAC-Ermittlung in `client_add.pl` erfolgen, so wäre ein Aufruf dieses Skripts nur dann möglich, wenn sich für den Client gegenwärtig ein Eintrag in der ARP-Tabelle befindet. Der Aufruf von `client_add.pl` soll aber einem Administrator zu jedem Zeitpunkt ohne Probleme möglich sein, deshalb scheidet diese Variante aus.

Stattdessen erwartet `client_add.pl` die MAC-Adresse als Parameter, und bei einer Nutzeranmeldung über das WWW-Formular wird die MAC im CGI-Skript ermittelt. Ein Administrator muß die MAC-Adresse gegebenfalls selbst ermitteln und als Parameter angeben. Diese Vorgehensweise erscheint dem Autor sinnvoller, als die MAC-Adresse zunächst in die ARP-Tabelle einzutragen und sie dann von `client_add.pl` wieder dort auslesen zu lassen.

Erreichbarkeitsprüfung

Ob zu einem Client noch Kontakt besteht, wird hier aktiv mit `ping` geprüft. Als Alternative käme auch eine passive Prüfung in Frage, zum Beispiel durch Auswertung der ARP-Tabelle [47], der Zähler in den Firewall-Regeln oder durch Zusammenarbeit mit dem DHCP-Server [48]. Als Kriterien für die automatische Abmeldung würde dann das Verschwinden des Eintrags aus der ARP-Tabelle, das Ausbleiben von Änderungen der Firewall-Zähler oder die Freigabe der DHCP-Lease dienen.

4.4.7 Sicherheitsfragen

Root-Privilegien und `sudo`

Aus Sicherheitsgründen läuft der WWW-Server nicht mit Root-Rechten, sondern unter einem eigenen Account. Zur eigentlichen Registrierung eines Nutzers sind aber Root-Privilegien erforderlich, um die Firewall-Regeln einzurichten. Die Nutzerumschaltung auf *root* ist hier mit `sudo` realisiert. Damit läßt sich detailliert angeben, welches Programm mit erweiterten Privilegien arbeiten soll und welcher Nutzer es so ausführen darf. Bei Anwendung des Setuid-Bits wäre die Rechtevergabe schwieriger, außerdem erlauben aktuelle UNIX-Systeme aus Sicherheitsgründen die Setuid-Ausführung von Skripten nicht mehr.

Taint Mode

Das CGI-Skript für die WWW-Anmeldung nutzt den *Taint Mode* von Perl, wie in der Perl-Dokumentation `perlsec(1p)` empfohlen. In diesem Modus aktiviert Perl zusätzliche Sicherheitsmechanismen, um zu verhindern, daß „unsichere“ Daten wie zum Beispiel Nutzereingaben ungeprüft verwendet werden. Falls der Programmierer vergißt, eine Eingabe zu prüfen, bricht das Skript die Ausführung ab, anstatt mit den potentiell gefährlichen Daten weiterzuarbeiten.

Verschlüsselung der Datenübertragung

Alle für die Anmeldung relevanten Daten, insbesondere das Paßwort, werden verschlüsselt übertragen (siehe Abschnitt 4.2.1). Die Übertragung der Nutzdaten nach der Anmeldung erfolgt dagegen unverschlüsselt; die Nutzer sind selbst dafür verantwortlich, ihre Übertragungen angemessen zu schützen. Das ist ein Kompromiß zwischen einfacher Nutzung des Systems und akzeptabler Sicherheit.

Mißbrauch des Funknetzes als Übertragungsweg

Die Access Points an den verschiedenen Standorten der TU Chemnitz sind in einem *Virtual LAN (VLAN)* zusammengefaßt. Dieses ist durch den Firewall vom Campusnetz getrennt. Innerhalb des VLANs existieren aber keine Beschränkungen, weder auf den Access Points noch dazwischen. Dadurch ist es möglich, daß zwei Rechner ohne Anmeldung innerhalb des Funknetzes kommunizieren, dies wird durch die vorliegende Firewall-Lösung nicht verhindert. Besonders wahrscheinlich ist dieser Mißbrauch bei räumlich weit auseinanderliegenden Access Points.

Eine Verbesserungsmöglichkeit könnte darin bestehen, die Access Points eines TU-Standorts jeweils in einem VLAN zusammenzufassen, und alle diese VLANs dann an den Firewall-Rechner heranzuführen. Dafür muß dieser

Rechner mit *VLAN tags* nach IEEE 802.1Q umgehen können oder entsprechend viele physische Netzwerkinterfaces besitzen. Außerdem sind Anpassungen an der Firewall-Software nötig, damit sie mehrere Subnetze unterstützt.

Race Condition

Im Zeitfenster zwischen dem Abschicken des Anmeldeformular durch den Nutzer und der Ermittlung der MAC-Adresse durch das CGI-Skript könnte ein Angreifer die MAC-Adresse im ARP-Cache des Firewalls durch seine eigene ersetzen, indem er die IP-Adresse des rechtmäßigen Nutzers bei sich konfiguriert. Es wird dann der Angreifer mit der „gestohlenen“ IP-Adresse angemeldet, nicht der rechtmäßige Nutzer. Die Beseitigung dieser Race Condidion wäre möglich, indem bei der Anmeldung die MAC-Adresse im Formular mit abgefragt wird. Allerdings leidet darunter die Benutzerfreundlichkeit.

Der Angriff ist relativ aufwendig durchzuführen, deshalb kann man davon auszugehen, daß ein Angreifer eher MAC- und IP-Adresse fälschen wird, als den Aufwand in Kauf zu nehmen. Selbst wenn der Angriff verhindert würde, wäre durch die gefälschte IP-Adresse die Netzverbindung des rechtmäßigen Nutzers gestört. Der Nutzen zusätzlicher Sicherheitsmaßnahmen erscheint deshalb fraglich, und es wurde hier zugunsten der Benutzerfreundlichkeit entschieden.

4.4.8 Prozesse und Interprozeßkommunikation

Bei der Implementierung wurde weitgehend auf selbstentwickelte Daemon-Prozesse verzichtet (Ausnahme: `dhcp-config-watcher.pl`), um die Robustheit der Software zu erhöhen. Wenn eine Fehlerbedingung unbehandelt bleibt und dadurch zum Abbruch eines Prozesses führt, hat das bei einem Daemon-Prozeß einen längeren Ausfall zur Folge, bis ein Administrator eingreift. Bei einem kurzlebigen Prozeß kann die fehlgeschlagene Aktion meist einfach wiederholt werden.

Die Datenspeicherung und die Kommunikation zwischen den Prozessen erfolgt mit Textdateien (siehe Abschnitt 4.5.3), die bei Bedarf für exklusiven Zugriff gesperrt werden. Es wäre auch möglich gewesen, dafür ein Datenbankmanagementsystem (DBMS) einzusetzen. Im Nachhinein betrachtet hätte sich der Aufwand gelohnt, insbesondere wären dann nachträgliche Änderungen einfacher zu realisieren.

4.4.9 Performance und Optimierungsmöglichkeiten

Zur Zeit melden sich pro Tag etwa 10 Nutzer an, die Zahl der gleichzeitigen Nutzer ist meist geringer als 3. Bisher existieren also noch keine Erfahrungswerte für den Fall, daß eine größere Zahl von Nutzern gleichzeitig aktiv ist.

Mit steigender Nutzerzahl erhöht sich sowohl das zu übertragende Datenvolumen als auch die Last durch Verwaltungsaufgaben. Zum Vergleich: Im Chemnitzer Studentennetz (CSN) bedient ein Server mit ähnlicher Leistungsfähigkeit etwa 1600 Nutzer und transportiert dabei ein Datenvolumen von bis zu 2,5 MByte pro Sekunde. Es existieren dort ebenfalls mehrere Firewall-Regeln pro IP-Adresse. Zumindest das Routing und ein zustandsloser Paketfilter sollten also keinen Engpaß darstellen.

Der Einsatz von NAT erfordert allerdings eine Buchführung über einzelne Verbindungen und benötigt wesentlich mehr CPU-Zeit und Speicher. Als Abhilfe könnte man zwar das 1:1 NAT durch Policy-Routing ersetzen, aber auf das n:1 NAT zur Umleitung unbekannter Clienten läßt sich nicht ohne weiteres verzichten, also bleibt das Connection Tracking immer noch nötig. Eventuell könnte die Umleitung noch per DNS-Server erreicht werden, indem dieser alle Anfragen mit der IP-Adresse des Firewalls beantwortet. Das hat allerdings den Nachteil, daß allen Clienten nach der Anmeldung ein neuer DNS-Server per DHCP zugewiesen werden muß. Außerdem könnten sich die falschen DNS-Informationen noch einige Zeit im DNS-Cache des Clienten befinden.

Einen weiteren Engpaß könnte das DHCP-Protokoll darstellen. Zur Zeit ist die maximale Gültigkeitsdauer der DHCP-Lease auf 1 Minute eingestellt, um eine schnelle Umkonfigurierung der Clienten zu ermöglichen. Das bedeutet aber auch, daß jeder angemeldete Rechner ungefähr aller 30 Sekunden die Gültigkeit seiner IP-Adresse verlängert, was sowohl im Funknetz als auch auf dem Firewall Last verursacht. Man könnte die maximale Gültigkeit zwar verlängern, nimmt dann aber längere Wartezeiten für die Nutzer in Kauf, es sei denn, sie führen die DHCP-Aktualisierung manuell durch.

Schließlich ist auch die CPU-Last durch die Erreichbarkeitsprüfung nicht zu vernachlässigen. Für jede Anfrage wird ein Kindprozeß gestartet, zur Zeit einmal pro Minute und Client. Ein Versuch hat ergeben, daß sich damit etwa 500 Anfragen pro Minute zuverlässig bewältigen lassen, wenn sonst keine Last vorhanden ist. Sollte diese Zahl von Clienten einmal erreicht werden, so könnte man das Abfrageintervall etwas vergrößern, oder das ping selbst implementieren anstatt Kindprozesse dafür zu starten. Es ist allerdings zu vermuten, daß die beiden erstgenannten Probleme wesentlich stärker in Erscheinung treten und Optimierungen eher dort nötig sind.

4.5 Administration

Dieser Abschnitt beschreibt das Softwarepaket aus der Sicht des Firewall-Administrators. Für Nutzer sollte die Software nahezu selbsterklärend sein, eine kurze Anleitung findet sich im Anhang B.

4.5.1 Systemvoraussetzungen und Installation

Die Software liegt als Paket `wlan-firewall` im RPM-Format vor (Redhat Package Manager Version 3). Sie wurde unter Redhat Linux 7.1.94 getestet und ist jetzt auch auf diesem System im Einsatz. Sie benötigt einen Linux-Kernel 2.4.x mit Unterstützung für *iptables* und die dazugehörigen Programme. Weiterhin sind ein Perl-Interpreter und das Paket `perl-Time-HiRes` aus den Redhat-Powertools erforderlich.

Bei der Installation wird zunächst das Start-Skript `/etc/rc.d/init.d/wlan-firewall` mit `chkconfig` in die gewünschten Runlevels (hier 2, 3, 4 und 5) eingebunden. Falls es sich um die Erstinstallation handelt, werden einige Initialisierungen durchgeführt:

- die Log-Datei `/var/wlan-firewall/anmeldung.log` wird angelegt und ihre Zugriffsrechte sowie ihr Eigentümer gesetzt
- die Liste der verfügbaren offiziellen IP-Adressen wird aus der Konfigurationsdatei `/etc/wlan-firewall/FW_Config.pm` erzeugt und in die Datei `/var/wlan-firewall/ips-public` geschrieben
- die DHCP-Konfigurationsdatei `/etc/dhcpd.conf` wird aus `/etc/dhcpd.conf.fixed` erzeugt; eine eventuell schon vorhandene DHCP-Konfiguration wird überschrieben

Wenn die Datei `ips-public` bereits vorhanden ist, werden diese Initialisierungsschritte übersprungen. In jedem Fall wird anschließend das Start-Skript aufgerufen, um die Software in Betrieb zu nehmen.

Die Deinstallation erfolgt ähnlich, dabei wird die Software beendet und die symbolischen Links wieder aus den Runlevels entfernt.

4.5.2 Systemstart und Herunterfahren

Beim Systemstart und beim Herunterfahren wird jeweils das Skript `/etc/init.d/wlan-firewall` mit dem Parameter `start` bzw. `stop` aufgerufen. Ein manueller Aufruf durch den Administrator ist ebenfalls möglich. In Abhängigkeit vom Parameter wird dann entsprechend die Funktion `start()` bzw. `stop()` im Skript verwendet.

Die Funktion `start()` lädt zunächst zwei Kernelmodule, um die Nutzung von FTP unter Einwirkung von NAT zu ermöglichen. Sie bringt die Firewall-Regeln auf die Standardeinstellung (keine Regeln, alles erlaubt) und setzt dann grundlegende Firewall-Regeln, die unabhängig von An- und Abmeldungen immer existieren. Dazu gehören u. a. die Rechte für die einzelnen Nutzerklassen. Die zu ladenden Firewall-Regeln sind in Dateien im Verzeichnis `/etc/wlan-firewall/start` abgelegt. Welche dieser Dateien geladen werden sollen und in welcher Reihenfolge, läßt sich mit der Variable `%templates` in `/etc/wlan-firewall/FW_Config.pm` konfigurieren.

Weiterhin löscht `start()` alle noch vorhandenen Registrierungen und erzeugt die Datei `/etc/dhcpd.conf` wie bei der Neuinstallation. Anschließend wird der Daemon-Prozeß `dhcp-config-watcher.pl` ausgeführt, welcher den DHCP-Server immer dann neu startet, wenn sich dessen Konfiguration geändert hat. Zum Schluß wird noch das Routing aktiviert. Damit ist die Software betriebsbereit.

Die Funktion `stop()` deaktiviert zuerst das Routing. Sie bringt die Firewallregeln auf die Standardeinstellung und installiert dann die Regeln aus dem Verzeichnis `/etc/wlan-firewall/stop`, zur Zeit ist dieses aber leer. Schließlich beendet sie noch den Prozeß `dhcp-config-watcher.pl`.

4.5.3 Dateien

WWW-Interface

Als Startseite dient die Datei `anmeldung.html`, sie enthält das HTML-Anmeldeformular. Die Formulardaten werden vom CGI-Skript `anmeldung.cgi` bzw. `anmeldung-auth.cgi` weiterverarbeitet. Was dabei im einzelnen abläuft, wurde bereits in Abschnitt 4.3.1 beschrieben.

Ausführbare Dateien

Für den Administrator existiert ein Kommandozeileninterface, um Clienten manuell zu (de)registrieren. Dafür sind die Skripte `client_add.pl` und `client_del.pl` zuständig. Ihre Funktionsweise ist in den Abschnitten 4.3.1 und 4.3.2 beschrieben.

Weiterhin existieren Kommandos, um Änderungen der offiziellen und privaten IP-Bereiche in Kraft zu setzen. Bei Änderungen an den privaten IP-Bereichen oder an der DHCP-Konfiguration muß die Datei `/etc/dhcpd.conf` neu erzeugt werden, das geschieht durch einen Aufruf von `update-dhcp.pl`.

Eine Liste der verfügbaren offiziellen IP-Adressen wird in `ips-public` verwaltet. Bei Änderungen an den offiziellen IP-Bereichen muß diese Liste neu erstellt werden, dazu dient `init-public-ips.pl`. Beim Aufruf dieses Skripts dürfen keine Clienten der Nutzerklassen `private` oder `public` registriert sein. Es empfiehlt sich deshalb, die Software vorher zu beenden und anschließend wieder zu starten, siehe Abschnitt 4.5.2.

Ähnliches gilt für `update-dhcp.pl`. Wenn sich allerdings nur die DHCP-Konfiguration ändert, ist kein Neustart der kompletten Firewall-Software nötig, es genügt der automatisch durchgeführte Neustart des DHCP-Servers.

Wie in Abschnitt 4.5.1 beschrieben, werden bei der Erstinstallation einige Initialisierungen durchgeführt. Dazu dienen die Skripte `init-logfile.pl`, `init-public-ips.pl` und `update-dhcp.pl`. Die beiden letztgenannten wurden im vorigen Teilabschnitt besprochen. Das Skript `init-logfile.pl` erzeugt die Datei `anmeldung.log`, falls sie noch nicht

existiert, setzt ihren Eigentümer auf `apache`, die Eigentümergruppe auf `adm` und setzt die Zugriffsrechte auf Lesen/Schreiben für den Eigentümer und Lesen für die Gruppe.

Die Funktionsweise des Startskripts `/etc/rc.d/init.d/wlan-firewall` wurde in Abschnitt 4.5.2 beschrieben. Auf dem hier eingesetzten Redhat-System kann ein Administrator das Hilfsprogramm `service` nutzen, um das Skript aufzurufen, z. B. `service wlan-firewall stop`. Das Startskript ruft dann weitere Skripte auf, um seine Aufgaben zu erfüllen. Dazu gehören `firewall_start.pl` und `firewall_stop.pl` zum Einrichten grundlegender Firewall-Regeln, das Skript `init-clients.pl` zum Initialisieren der Datei mit den akuell angemeldeten Clienten, und schließlich das Skript `dhcp-config-watcher.pl`, welches als Daemon für den Neustart des DHCP-Servers sorgt. `dhcp-config-watcher.pl` verhindert die gleichzeitige Ausführung mehrerer Neustart-Anforderungen und erzwingt eine Mindestpause zwischen den Neustarts.

Wie weiter oben bereits dargestellt, sollen periodisch wiederkehrende Aufgaben möglichst nicht von einem selbstentwickelten Daemon behandelt werden. Als Alternative wird hier der bewährte `crond` eingesetzt, um die Skripte `pinger.pl` und `get_zin.pl` regelmäßig auszuführen. Beide Skripte kann auch ein Administrator von Hand aufrufen, das ist aber normalerweise nicht nötig.

`get_zin.pl` sorgt für die Aktualisierung der ZIN-Informationen (Abschnitt 4.3.4) und wird stündlich ausgeführt. Der Ausführungszeitpunkt liegt dabei kurz nach der Erzeugung der Quelldateien, um eine möglichst hohe Aktualität zu gewährleisten.

Das Skript `pinger.pl` prüft die Erreichbarkeit der angemeldeten Clienten (Abschnitt 4.3.2) und läuft jede Minute. Das Intervall zwischen zwei Ping-Anfragen an einen Client-Rechner läßt sich mit der Variable `$ping_host_interval` in `FW_Config.pm` vergrößern. Das Aufrufintervall von `pinger.pl` sollte trotzdem auf eine Minute eingestellt bleiben.

Die Datei `FW_Utils.pm` ist selbst nicht ausführbar, sie enthält eine Sammlung von Perl-Funktionen und wird von den meisten der Perl-Skripte genutzt.

Konfiguration

`/etc/wlan-firewall/FW_Config.pm` ist die zentrale Konfigurationsdatei. Sie enthält Angaben zu Dateipfaden, Nutzerklassen, Netzwerkinterfaces und zu den Dateien mit Firewall-Regeln, die Ping-Konfiguration sowie die Bereiche der offiziellen und privaten IP-Adressen.

Die Datei `/etc/dhcpd.conf.fixed` enthält die unveränderlichen Bestandteile der DHCP-Konfiguration sowie Variablen als Platzhalter für diejenigen Teile, die erst bei Client-Anmeldungen hinzukommen oder die aus `FW_Config.pm` ermittelt werden. `/etc/dhcpd.conf.fixed` dient als Vorlage,

aus der dann die eigentliche DHCP-Konfigurationsdatei `/etc/dhcpd.conf` erzeugt wird.

`/etc/wlan-firewall/add_static_clients` ist ein ausführbares Shell-Skript und enthält Befehle, um statische Clienten wie Access Points beim Systemstart zu registrieren. Es ist hier trotzdem als Konfigurationsdatei aufgeführt, weil es zur Bearbeitung durch den Administrator vorgesehen ist.

Schließlich sind auch die Firewallregeln in den Verzeichnissen `start/`, `stop/` und `client/` unterhalb `/etc/wlan-firewall/` konfigurierbar. Die Dateien in `/etc/wlan-firewall/start/` werden beim Systemstart durch `firewall_start.pl` geladen, analog gilt das beim Herunterfahren für `/etc/wlan-firewall/stop/` und `firewall_stop.pl`. Die Dateien darin enthalten jeweils statische Regeln, die sich bei An- und Abmeldungen von Clienten nicht ändern, zum Beispiel zur Festlegung der Nutzerklassen-Rechte. Im Verzeichnis `/etc/wlan-firewall/client/` befinden sich dagegen die client-spezifischen Firewall-Regeln, zum Beispiel zur Einordnung der IP-Datagramme in die Nutzerklasse oder für 1:1 NAT. Diese Dateien werden bei der An- oder Abmeldung eines Clienten ausgewertet.

Wenn in einem der Verzeichnisse eine Datei hinzugefügt oder entfernt werden soll, so muß das außerdem in der Variable `%templates` in `FW_Config.pm` geändert werden. Um eine der Dateien zeitweise zu deaktivieren, genügt sogar das Ändern dieser Variable; es ist nicht nötig, die Datei selbst zu löschen.

Das Format der Dateien entspricht der Ausgabe von `iptables-save`, letztendlich wird zum Laden der Firewall-Regeln `iptables-restore` verwendet. Zusätzlich können die Dateien Variablen enthalten, die vor dem Laden der Firewall-Regeln durch ihre aktuellen Werte ersetzt werden. Einige Variablen sind allgemeingültig und in allen 3 Verzeichnissen nutzbar (Tabelle 4.1), andere sind client-spezifisch und deshalb nur im Verzeichnis `client` erlaubt (Tabelle 4.2).

Variable	**Beschreibung**
`$action`	`-A` (Regel hinzufügen) oder `-D` (Regel löschen)
`$iface_int`	Name des Netzwerkinterfaces auf Wireless-LAN-Seite
`$iface_ext`	Name des Uplink-Interfaces
`$IP_iface_ext`	IP-Adresse des Uplink-Interfaces
`$guest`, `$private`, `$public`, `$static`	numerische ID zur Nutzerklasse (`$guest`=1, `$private`=2, etc.)

Tabelle 4.1: Allgemeingültige Firewall-Variablen

Variable	Beschreibung
`$hwaddr`	MAC-Adresse des Clienten
`$IP_int`	IP-Adresse im Wireless LAN nach der Anmeldung
`$IP_ext`	im Internet sichtbare IP-Adresse des Clienten (oder „-“ wenn sie gleich `$IP_int` ist)
`$userclass`	Bezeichnung der Nutzerklasse
`$userclass_num`	numerische ID der Nutzerklasse

Tabelle 4.2: Clientspezifische Firewall-Variablen

Dateien im Verzeichnis `/var/wlan-firewall/`

Die Datei `ips-public` enthält eine Liste der verfügbaren offiziellen IP-Adressen und ist als Warteschlange (FIFO) organisiert. Bei der Anmeldung von Clienten wird bei Bedarf am Anfang der Datei eine IP-Adresse entnommen und bei der Abmeldung wieder am Ende angefügt. Die Datei liegt im Textformat vor und enthält eine IP-Adresse pro Zeile.

`clients` ist ebenfalls eine Textdatei und enthält pro Zeile einen registrierten Clienten. Die Bedeutung der Spalten ist in Tabelle 4.3 angegeben.

#	Name	Beschreibung
1	`hwaddr`	
2	`userclass`	gleiche Bedeutung wie die entsprechenden
3	`IP_int`	Variablen in Tabelle 4.2
4	`IP_ext`	
5	`pings_lost`	Anzahl der Ping-Anfragen an diesen Client, auf die der Firewallrechner keine Antwort erhalten hat
6	`last_changed`	Zeitpunkt der letzten Änderung an diesem Eintrag in Sekunden seit 1.1.1970 0:00 UTC

Tabelle 4.3: Bedeutung der Spalten in der Datei `clients`

In der Datei `anmeldung.log` werden An- und Abmeldungen vermerkt, sowohl bei Erfolg als auch bei aufgetretenen Fehlern. Die Anmeldedaten werden vom CGI-Skript geschrieben, bei der Abmeldung ist das Skript `client_del.pl` dafür verantwortlich.

Die Datei `HOME-PAGE` enthält die Kennzeichen aller URZ-Nutzer, die ein ZIN besitzen oder davon befreit sind. Diese sind berechtigt, sich mit Nutzerkennzeichen und Paßwort in den Nutzerklassen `private` und `public` anzumelden, allen anderen Nutzern steht nur der Gast-Zugang zur Verfügung, siehe auch Abschnitt 4.3.4.

Cron-Jobs

Auf Seite 40 wurden die Cron-Jobs `get_zin.pl` und `pinger.pl` beschrieben. Ihre Einbindung in die Cron-Konfiguration erfolgt über die Datei `/etc/cron.d/wlan-firewall`. Dort sind die Ausführungszeitpunkte und -intervalle festgelegt: minütlich für `pinger.pl` und jeweils zehn Minuten vor der vollen Stunde (:50) für `get_zin.pl`.

4.5.4 Standardsoftware und deren Konfiguration

In den Abschnitten 4.2.1 bis 4.2.3 wurden bereits einige zusätzlich benötigte Softwarepakete erwähnt. Dieser Abschnitt enthält nun weitere Details zur Konfiguration.

DNS-Server

Zum Einsatz kommt hier die Software Bind Version 9.1.3 vom Internet Software Consortium. Die Proxy-Funktion für den Zugriff auf die weltweiten DNS-Informationen ist folgendermaßen realisiert (Auszug aus `named.conf`):

```
options {
        forwarders {
                134.109.132.51;
                134.109.132.55;
        };
};
```

Damit werden alle Anfragen, die der DNS-Server nicht selbst beantworten kann, an die DNS-Server des URZ weitergeleitet.

Die Namensauflösung für die privaten IP-Adressen ist wie folgt konfiguriert (Auszug aus der Datei `named.conf`):

```
acl can_query {
        127.0.0.1/32;
        134.109.144.0/23;
        192.168.0.0/16;
};
zone "funklan.hrz.tu-chemnitz.de" IN {
        type master;
        file "db.funklan.hrz.tu-chemnitz.de";
        allow-query { can_query; };
};
zone "168.192.in-addr.arpa" IN {
        type master;
        file "db.168.192";
        allow-query { can_query; };
};
```

Die Option `allow-query` bewirkt dabei, daß der DNS-Server nur aus dem Funknetz erreichbar ist.

Die Zuordnung der IPs zu den Namen befindet sich in der Datei `db.funklan.hrz.tu-chemnitz.de`:

```
...
@               IN      NS      wfire-i
funk-netz       IN      A       192.168.0.0
funk-bcast      IN      A       192.168.255.255
wfire-i         IN      A       192.168.0.254

funk1           IN      A       192.168.0.1
funk2           IN      A       192.168.0.2
...
funk253         IN      A       192.168.0.253
; 192.168.0.254 reserviert für Gateway
funk255         IN      A       192.168.0.255
funk256         IN      A       192.168.1.0
funk257         IN      A       192.168.1.1
...
funk65533       IN      A       192.168.255.253
funk65534       IN      A       192.168.255.254
```

und das *reverse lookup* in der Datei `db.168.192`:

```
...
@       IN      NS      wfire-i.funklan.hrz.tu-chemnitz.de.
0.0     IN      PTR     funk-netz.funklan.hrz.tu-chemnitz.de.
255.255 IN      PTR     funk-bcast.funklan.hrz.tu-chemnitz.de.
254.0   IN      PTR     wfire-i.funklan.hrz.tu-chemnitz.de.

1.0     IN      PTR     funk1.funklan.hrz.tu-chemnitz.de.
2.0     IN      PTR     funk2.funklan.hrz.tu-chemnitz.de.
...
253.0   IN      PTR     funk253.funklan.hrz.tu-chemnitz.de.
; 192.168.0.254 reserviert für Gateway
255.0   IN      PTR     funk255.funklan.hrz.tu-chemnitz.de.
0.1     IN      PTR     funk256.funklan.hrz.tu-chemnitz.de.
1.1     IN      PTR     funk257.funklan.hrz.tu-chemnitz.de.
...
253.255 IN      PTR     funk65533.funklan.hrz.tu-chemnitz.de.
254.255 IN      PTR     funk65534.funklan.hrz.tu-chemnitz.de.
```

DHCP-Server

Die Konfiguration des DHCP-Servers erfolgt über die Datei `dhcpd.conf.fixed`, welche nachstehend abgebildet ist. Daraus generiert dann `update-dhcp.pl` die eigentliche DHCP-Konfigurationsdatei `dhcpd.conf`. Dabei werden `#PRIVATE_RANGES` und `#CLIENTS` ersetzt.

```
authoritative;
shared-network urz-wireless {
  subnet 134.109.144.0 netmask 255.255.255.0 {
    option subnet-mask 255.255.255.0;
    option broadcast-address 134.109.144.255;
    option routers 134.109.144.254;
    option domain-name-servers 134.109.144.254;
    option domain-name "hrz.tu-chemnitz.de";
  }
  subnet 192.168.0.0 netmask 255.255.0.0 {
    option subnet-mask 255.255.0.0;
    option broadcast-address 192.168.255.255;
    option routers 192.168.0.254;
    option domain-name-servers 192.168.0.254;
    option domain-name "hrz.tu-chemnitz.de";
    default-lease-time 60;
    max-lease-time 60;
    # automatically generated list of private IP ranges:
    # PRIVATE_RANGES
    # automatically generated list of private IP ranges (end)
  }
}
group {
  default-lease-time 120;
  max-lease-time 120;

  # automatically generated list of fixed address assignments:
  # CLIENTS
  # automatically generated list of fixed address
  # assignments (end)
}
```

Die Deklaration `shared-network` gibt an, daß die beiden Subnetze `134.109.144.0/24` und `192.168.0.0/16` auf dem gleichen Netzwerk arbeiten, in diesem Fall realisiert über ein Alias-Interface. Netzmaske, Broadcast-Adresse und Default-Router werden als Option an die Clienten übermittelt, ebenso der DNS-Server und der DNS-Domainname. Als Default-Router und DNS-Server ist dabei die IP-Adresse des Firewall-Rechners aus dem jewei-

ligen Subnetz angegeben. Für unbekannte Rechner und Clienten der Nutzerklasse `private` ist die Gültigkeitsdauer auf 1 Minute eingestellt, für die Nutzerklassen `public` und `static` auf 2 Minuten.

Der DHCP-Server soll ebenfalls nur aus dem Funknetz erreichbar sein, dazu wird er beim Start auf das interne Interface beschränkt. Auf einem Redhat-System ist für solche Optionen die Datei `/etc/sysconfig/dhcpd` vorgesehen:

```
# Command line options here
DHCPDARGS=eth1
```

HTTP-Server

Bei nicht gefundenen Seiten oder verweigertem Zugriff liefert der HTTP-Server statt einer Fehlermeldung eine Umleitung auf die Anmeldeseite. In der Konfigurationsdatei `httpd.conf` sieht das folgendermaßen aus:

```
# Customizable error response (Apache style)
# not found:
ErrorDocument 404 \
  https://wfire.hrz.tu-chemnitz.de/anmeldung.html
# Permission denied:
ErrorDocument 403 \
  https://wfire.hrz.tu-chemnitz.de/anmeldung.html
```

Die Angabe eines absoluten URL inklusive Hostname des Servers bewirkt, daß der Client eine HTTP-Umleitung geschickt bekommt und dann selbst die Anmeldeseite abruft. Wäre stattdessen nur ein Dateipfad angegeben, würde der Server die Anmeldeseite sofort unter dem falschen URL ausliefern. Die Anmeldeseite muß aber unter ihrem richtigen URL abgerufen werden, damit die Prüfung des SSL-Zertifikats funktioniert und damit die Statusseite nach der Anmeldung korrekt übertragen werden kann, obwohl dann das DNAT bereits deaktiviert ist.

Die Zugriffrechte auf die Anmeldeseiten sind wie folgt konfiguriert:

```
Order allow,deny
Allow from 192.168.0.0/16
```

In Verbindung mit einer Route für 192.168.0.0/16 auf das interne Interface und aktiviertem *reverse path filter* sind die Seiten nur noch aus dem Funknetz erreichbar.

Mit der Option `SSLRequireSSL` wird festgelegt, daß das CGI-Skript nur über eine verschlüsselte Verbindung benutzt werden darf. Wenn das Skript unter dem Name `anmeldung-auth.cgi` abgerufen wird, muß sich der Nutzer mit seinem URZ-Paßwort authentifizieren:

```
<Files anmeldung-auth.cgi>
  AuthType Basic
  AuthName "URZ-Login"
  require valid-user
</Files>
```

Die Option `-Indexes` verbietet global das Auflisten von Verzeichnisinhalten, aufgrund der Umleitung im Fehlerfall wird dann wieder die Anmeldeseite ausgeliefert.

Im Anmeldeformular muß sichergestellt werden, daß das CGI-Skript immer per SSL aufgerufen wird, sogar dann, wenn das Anmeldeformular selbst unverschlüsselt übertragen wurde. Zum Umschalten auf HTTPS kann man einen absoluten URL verwenden, muß dann aber den Servername mit angeben. Hier kommt stattdessen das Apache-Modul `mod_rewrite` zum Einsatz, damit ist die Umschaltung auf SSL auch mit einem URL der Art `/cgi-bin/anmeldung-auth.cgi:SSL` möglich [49]. Die Konfiguration für `mod_rewrite` lautet:

```
RewriteEngine on
RewriteRule   ^/(.*):SSL$   https://%{SERVER_NAME}/$1 [R,L]
RewriteRule   ^/(.*):NOSSL$ http://%{SERVER_NAME}/$1  [R,L]
```

Für die Authentifizierung wird das Apache-PAM-Modul `mod_auth_pam` verwendet:

```
LoadModule pam_auth_module    modules/mod_auth_pam.so
```

Dieses wiederum greift über das Modul `pam_afs.so` auf den AFS/Kerberos-Server zurück (`/etc/pam.d/httpd`):

```
# The PAM configuration file for the 'httpd' service
auth       sufficient /lib/security/pam_afs.so.1 \
                              ignore_root dont_fork
auth       required   /lib/security/pam_pwdb.so
session    optional   /lib/security/pam_afs.so.1
session    required   /lib/security/pam_pwdb.so
account    required   /lib/security/pam_pwdb.so
```

Eine Besonderheit an der TU Chemnitz besteht darin, daß die meisten Nutzer ihre HTTP-Proxy-Einstellungen automatisch über die Datei `http://www.tu-chemnitz.de/misc/proxy.proxy` beziehen. Die bisher beschriebene Konfiguration würde dazu führen, daß unter diesem URL die Anmeldeseite ausgeliefert wird. Der Browser würde das als Fehler melden, und eventuell wäre dann auch die Proxy-Konfiguration falsch. Diese Probleme lassen sich vermeiden, indem der HTTP-Server die Datei `/misc/proxy.proxy` mit dem richtigen MIME-Typ bereitstellt:

```
<Location /misc>
AddType application/x-ns-proxy-autoconfig .proxy
</Location>
```

Die Datei ist keine Kopie vom offiziellen WWW-Server der TU, sondern es wird über einen symbolischen Link direkt die offizielle Version verwendet, um die Aktualität sicherzustellen.

Kapitel 5

Fazit und Perspektiven

5.1 Bewertung

Mit der vorliegenden Software wurde ein System realisiert, das den Wireless-LAN-Zugang zum Campusnetz und zum Internet automatisch verwaltet. Für die Anmeldung genügt ein WWW-Browser und ein DHCP-Client, beides ist auf den meisten Rechnern bereits vorhanden. Nach der Anmeldung erfolgt der Zugang transparent wie beim Anschluß an eine Ethernet-Dose, es ist keine besondere Kommunikationssoftware erforderlich.

Das System unterscheidet zwischen Gastnutzern mit eingeschränkten Rechten und authentifizierten URZ-Nutzern mit vollen Rechten. Letztere werden aus technischen Gründen noch einmal in zwei Klassen unterteilt, wobei eine der beiden Klassen (`private`) dafür sorgt, daß sich bei der Anmeldung die IP-Adresse nicht ändert, während die andere Klasse (`public`) durch den Verzicht auf NAT volle Transparenz bei der Datenübertragung gewährleistet.

Die Firewall-Regeln stellen sicher, daß eine unbefugte Nutzung des Netzzugangs deutlich erschwert ist, ohne die rechtmäßige Nutzung zu beeinträchtigen. Die Anmeldedaten werden verschlüsselt übertragen, um sie gegen Abhören und Mißbrauch zu schützen. Die eigentliche Datenübertragung erfolgt unverschlüsselt. Stattdessen wird den Nutzern empfohlen, selbst eine angemessene Verschlüsselung einzusetzen. Alternativen mit integrierter Verschlüsselung der Nutzdaten wurden untersucht, aber wegen mangelhafter Schutzwirkung oder hohem Aufwand für die Nutzer verworfen. Der Vorteil ist eine unkomplizierte Nutzung des Zugangs, ohne daß wesentliche Eingriffe auf dem mobilen Rechner nötig sind.

5.2 Entwicklungsmöglichkeiten

5.2.1 Gastzugang mit erweiterten Rechten

Die Rechte beim Gastzugang sind bei der vorliegenden Software stark eingeschränkt, es ist nur der Zugriff auf WWW-Server der TU Chemnitz erlaubt. Für spezielle Anwendungsfälle wie Konferenzen und öffentliche Veranstaltungen wären zusätzliche Nutzerklassen wünschenswert, die auch Zugriff auf Server außerhalb der Universität bieten und trotzdem kein URZ-Nutzerkennzeichen oder ZIN erfordern.

Als Zugangsschutz könnte ein Kennwort dienen, das nur für die Dauer der Veranstaltung gültig bleibt und allen Teilnehmern mitgeteilt wird. Welche Nutzungsarten gegenüber dem normalen Gastzugang zusätzlich erlaubt werden, muß sich an der voraussichtlichen Vertrauenswürdigkeit der Veranstaltungsteilnehmer orientieren. Die Implementierung dieser Variante sollte mit relativ geringem Aufwand möglich sein.

Eine andere Möglichkeit wäre eine Art „Bürgschaft“ durch einen URZ-Nutzer, der sich gegenüber dem System authentifiziert und die Gastanmeldung vorbereitet, zum Beispiel durch Setzen eines Einmalpaßworts oder eine zeitlich befristete Registrierung der MAC-Adresse des Gastrechners. Diese Variante ist etwas aufwendiger zu implementieren. Vermutlich würde sich in einem solchen Fall aber der URZ-Nutzer direkt anstelle des Gastes anmelden und diesem seinen vollwertigen Internetzugang zur Verfügung stellen. Mit anderen Worten: Dem Autor erscheint die Akzeptanz eines solchen Systems fraglich.

5.2.2 Unterstützung für mehrere Funk-Subnetze

In Abschnitt 4.4.7 wurde eine Möglichkeit beschrieben, wie das Funknetz von Außenstehenden als Übertragungsweg mißbraucht werden kann, und eine Lösungsvariante dafür aufgezeigt. Die Anpassung der Firewall-Software an mehrere Subnetze/VLANs könnte also eine sinnvolle Erweiterung darstellen.

5.2.3 Mobile IP

Die automatische Konfiguration der Mobilrechner durch DHCP ist meist akzeptabel. Falls doch einmal mit der ursprünglichen Netzwerkkonfiguration gearbeitet werden soll, damit der Mobilrechner logisch zum Heimatnetz gehört, würde sich der Einsatz von Mobile IP [50] empfehlen. Dafür wären mindestens folgende Vorkehrungen nötig:

- Betrieb eines *home agent* im Heimatnetz des Mobilgeräts
- wenn Mobile IP über die TU Chemnitz hinaus benutzt werden soll, Betrieb (mindestens) eines *foreign agent* im Funknetz

- Installation der Mobile-IP-Software auf dem Mobilrechner
- für jeden Mobilrechner Konfiguration eines gemeinsamen geheimen Schlüssels mit dem Home Agent

Es ist anscheinend keine brauchbare Mobile IP Client-Implementierung für aktuelle Windows-Systeme verfügbar. Auf einer aktuellen Linux-Version (Kernel 2.4.2) hat sich lediglich Dynamics Mobile IP von der Technischen Universität Helsinki [51] als tauglich erwiesen.

Mobile IP kann deshalb nur eine Ergänzung zur beschriebenen Software sein, diese aber nicht ersetzen. Die Zielgruppe ist deutlich kleiner. Aufgrund des hohen organisatorischen Aufwandes wurde Mobile IP im Rahmen dieser Arbeit nicht realisiert, das könnte aber Gegenstand einer weiteren Projektarbeit sein.

Anhang A

Administration: HOWTO

Bei allen hier beschriebenen Änderungen empfiehlt es sich, während der Wartungsarbeiten den HTTP-Server zu beenden, um Neuanmeldungen zu verhindern. Außerdem ist der Aufruf von `service wlan-firewall` mit dem Parameter `stop` bzw. `start` erforderlich.

A.1 Änderung von IP-Adressen oder Subnetzen

A.1.1 Firewall-IP auf dem externen Interface ändern

- neue IP-Adresse in der Datei
  `/etc/sysconfig/network-scripts/ifcfg-eth0`
  unter `IPADDR` eintragen
- Interface eth0 neu starten, um diese IP zu übernehmen
- in der Datei `/etc/wlan-firewall/FW_Config.pm` die neue IP-Adresse unter `$variables{"IP_iface_ext"}` eintragen
- in der gleichen Datei die Variable `@public_IP_ranges` überprüfen, die neue IP-Adresse darf in keinem der Adreßbereiche enthalten sein
- im URZ-Router die neue IP-Adresse als Gateway für das Funk-Subnetz eintragen
- im DNS-Server des URZ die Einträge für den Firewall-Rechner auf die neue IP-Adresse umstellen

A.1.2 Firewall-IP auf dem internen Interface ändern

- neue IP-Adresse in der Datei
  `/etc/sysconfig/network-scripts/ifcfg-eth1`
  unter `IPADDR` eintragen

- Interface eth1 neu starten, um diese IP zu übernehmen
- in der Datei `/etc/wlan-firewall/FW_Config.pm`
  die Variable `@private_IP_ranges` überprüfen,
  die neue IP-Adresse darf in keinem der Adreßbereiche enthalten sein
- im lokalen DNS-Server die Einträge für den Firewall-Rechner auf die neue IP-Adresse umstellen

A.1.3 Subnetz der offiziellen IP-Adressen ändern

- IP-Adresse des externen Firewall-Interfaces ändern, wenn nötig (siehe oben)
- offizielles IP-Subnetz in der Datei `/etc/dhcpd.conf.fixed` ändern
- `update-dhcp.pl` aufrufen
- in der Datei `/etc/wlan-firewall/FW_Config.pm` die Variable `@public_IP_ranges` ändern
- `init-public-ips.pl` aufrufen
- im URZ-Router das Funk-Subnetz ändern
- im DNS-Server des URZ die Einträge für das Funk-Subnetz ändern

A.1.4 Subnetz der privaten IP-Adressen ändern

- IP-Adresse des internen Firewall-Interfaces ändern, wenn nötig (siehe oben)
- privates Subnetz in der Datei `/etc/dhcpd.conf.fixed` ändern
- `update-dhcp.pl` aufrufen
- in der Apache-Konfigurationsdatei `/etc/httpd/conf/httpd.conf` die Klausel `allow from` auf das neue Subnetz ändern
- in der Datei `/etc/wlan-firewall/FW_Config.pm` die Variable `@private_IP_ranges` ändern
- im lokalen DNS-Server die Einträge für private IP-Adressen ändern

A.2 Hinzufügen von Nutzerklassen

- in der Datei `FW_Config.pm` Name und ID der neuen Nutzerklasse in die Variable `%userclasses` eintragen

- in der gleichen Datei in der Variable `%templates` festlegen, welche der Firewall-Dateien aus `/etc/wlan-firewall/client` für diese Nutzerklasse verwendet werden sollen

- im HTML-Anmeldeformular `/var/www/wfire/anmeldung.html` die Auswahl der neuen Nutzerklasse ermöglichen

- das CGI-Anmeldeskript `/var/www/cgi-bin/anmeldung.cgi` anpassen (siehe Sonderbehandlungen darin für einzelne Nutzerklassen)

- in der Datei `FW_Utils.pm` die Funktionen `needs_dhcp_entry()`, `get_ip_pub()` und `free_ip_pub()` überprüfen und evtl. anpassen

- Firewall-Datei `start/snat_out` anpassen, wenn die neue Nutzerklasse NAT verwendet (ähnlich `guest` oder `private`)

- Firewall-Dateien `client/classify_incoming_*` verwenden, wenn Rechner der neuen Klasse von außen unter einer eigenen IP-Adresse sichtbar sind (z. Zt. alle außer `guest`)

- in der Firewall-Datei `start/forward`:

  - Chains anlegen: `incoming_<klasse>` und `outgoing_<klasse>`
  - Firewall-Regeln in diese Chains einfügen, im einfachsten Fall eine allgemeine Regel mit `ACCEPT` oder `DROP`
  - in den Chains `incoming` und `outgoing` bei der Nutzerklassenprüfung auch die neue Nutzerklasse erkennen und die zugehörige Chain aufrufen, die weiter oben erzeugt wurde

- Hinweis: Die Daten von außen nach innen, d. h. vom Campusnetz oder Internet ins Funknetz, werden zur Zeit immer in die Nutzerklasse `guest` eingeordnet, wenn die Ziel-IP dem externen Interface des Firewall-Rechners entspricht (Antwortpakete in einer Datenübertragung mit N:1 NAT). Für eine detailliertere Bestimmung der Nutzerklasse eignet sich in diesem Fall die `FORWARD`-Chain besser als `PREROUTING`.

A.3 Anmeldung von Clienten der Nutzerklasse `static`

- bis zum nächsten Systemstart: Aufruf von
  `client_add.pl <Hardware-Adresse> static <aktuelle IP>`
- dauerhaft: obige Kommandozeile in die Datei
  `/etc/wlan-firewall/add_static_clients` eintragen

Anhang B

Nutzerdokumentation

B.1 Systemanforderungen

Für die Nutzung des Netzzugangs per Funk ist ein netzwerkfähiger Rechner mit Wireless-LAN-Interface, WWW-Browser und DHCP-Client erforderlich. Der Browser sollte Tabellen, Formulare und HTTP über SSL/TLS unterstützen. Für den DHCP-Client gibt bei den Windows-Systemen keine Wahlmöglichkeit, unter Linux stehen mehrere Implementierungen zur Verfügung: `pump`, `dhcp-client` aus der ISC-Distribution und `dhcpcd`.

Alle genannten DHCP-Clienten sind in der Lage, zur Erstkonfiguration des Netzwerkinterfaces eine IP-Adresse und die zugehörigen Parameter zu beschaffen. In der Nutzerklasse `public` muß der DHCP-Client aber außerdem auf Änderungen der IP-Adresse reagieren können. Zusätzlich wurde noch das Verhalten getestet, wenn das Verlängern der Lease fehlschlägt. In diesem Fall darf der Client die IP-Adresse nicht weiter benutzen. Die Testergebnisse im Überblick:

	Windows	pump	ISC	dhcpcd
IP-Änderung	+	-	+	-
Lease-Ablauf	-	-	-[1]	+

Tabelle B.1: DHCP-Clienten im Test

Die Problemfälle im Einzelnen:

- Wenn der DHCP-Server dem Client eine geänderte IP-Adresse zuteilt, beendet sich `pump` und die alte IP-Adresse bleibt aktiv. Der `dhcpcd` entfernt die alte IP-Adresse auf dem Interface, konfiguriert aber nicht die neue IP.

[1] Dieses Verhalten ist konfigurierbar.

- Wenn das Erneuern der Lease fehlschlägt, so nutzen Windows, `pump` und `dhcp-client` (ISC) die alte IP-Adresse weiter. Bei der ISC-Software läßt sich dieser Bug direkt im Konfigurationsskript `/etc/dhclient-script` beheben, oder auch separat in `/etc/dhclient-exit-hooks`:

```
if [ x$reason = xEXPIRE ] || [ x$reason = xFAIL ]; then
  if [ x$old_ip_address != x ]; then
    ifconfig $interface inet 0
  fi
fi
```

Unter Linux ist deshalb der `dhcp-client` vom ISC am besten geeignet.

B.2 Ablauf der Anmeldung

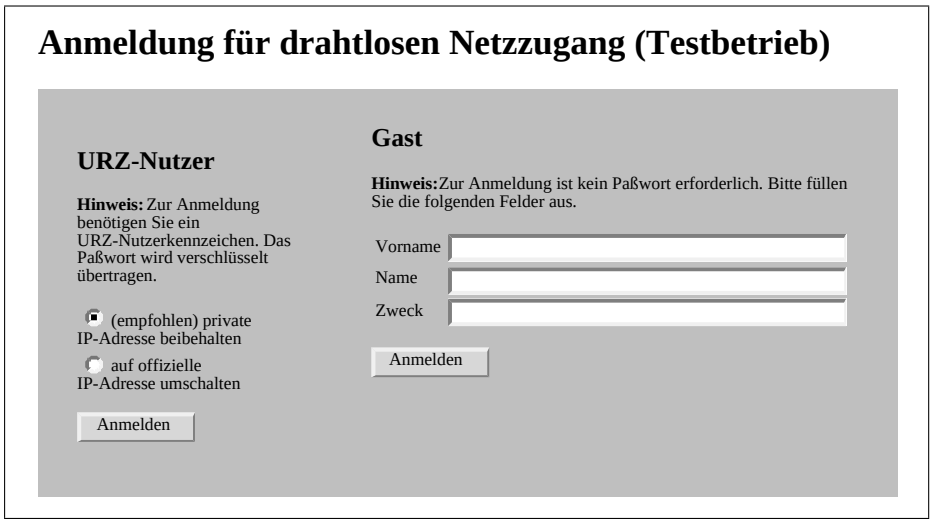

Abbildung B.1: Anmeldeformular

URZ-Nutzer wählen im linken Teil des Anmeldeformulars (Abbildung B.1) ihre Nutzerklasse aus. „Private IP-Adresse beibehalten“ entspricht der Nutzerklasse `private` und kommt ohne Änderung der IP-Adresse bei der Anmeldung aus. „Auf offizielle IP-Adresse umschalten“ enspricht der Klasse `public`. Hier ändert sich die IP-Adresse bei der Anmeldung, und der Nutzer muß dafür sorgen, daß sein DHCP-Client die neue IP-Adresse übernimmt, oder warten, bis das automatisch geschieht. Nach Auswahl der Nutzerklasse wird mit dem Knopf „Anmelden“ (links) die Anmeldung ausgelöst. Der WWW-Browser fragt nach URZ-Nutzerkennzeichen und AFS-Paßwort,

der Besitz des Zertifikats Internet-Nutzung (ZIN) wird automatisch geprüft. Anschließend wird eine Seite mit dem Status der Anmeldung (erfolgreich, fehlgeschlagen) ausgegeben.

Gäste tragen im rechten Teil ihren Namen und Vornamen sowie den Nutzungszweck ein. Der Knopf „Anmelden“ (rechts) löst dann die Anmeldung in der Nutzerklasse `guest` aus, danach wird ebenfalls eine Statusseite ausgegeben. Nutzerkennzeichen, Paßwort oder ZIN sind nicht erforderlich. Zur Zeit dürfen Gäste nur WWW-Seiten von Servern der TU Chemnitz abrufen.

Literaturverzeichnis

[1] Internet Software Consortium, *Internet Domain Survey: Number of Internet Hosts,* `http://www.isc.org/ds/host-count-history.html`

[2] GSM Association, *GSM Data Services,* `http://www.gsmworld.com/technology/data_services.html`

[3] Richard Sietmann, *Quo vadis, Mobilfunk? Nebenbuhler und Nachfolger von UMTS,* c't 5/2001, S. 94, Verlag Heinz Heise GmbH & Co KG

[4] UMTS Forum, *What is UMTS?,* `http://www.umts-forum.org/what_is_umts.html`

[5] Heinz Ochsner, DECT Forum, *DECT—The Standard explained,* `http://www.dect.ch/publicdocs/TechnicalDocument.PDF`

[6] Dušan Živadinović, *Kabelsalat ade: Schnurlos surfen,* c't 20/1998, S. 128, Verlag Heinz Heise GmbH & Co KG

[7] Institute of Electrical and Electronics Engineers, Inc. (IEEE), *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,* `http://standards.ieee.org/getieee802/802.11.html`

[8] Institute of Electrical and Electronics Engineers, Inc. (IEEE), *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Higher-Speed Physical Layer Extension in the 2.4 GHz Band,* `http://standards.ieee.org/getieee802/802.11.html`

[9] Gerrit Schulte, *Wellenreiter — Technik und Standardisierung von drahtlosen Netzen,* c't 6/1999, S. 222, Verlag Heinz Heise GmbH & Co KG

[10] Dušan Živadinović, *Privatfunk — Bluetooth als Netzwerk- und Schnittstellenmodul,* c't 25/1999, S. 126, Verlag Heinz Heise GmbH & Co KG

[11] Bluetooth Special Interest Group (SIG), `http://www.bluetooth.com/`

[12] Universität Rostock, FB Informatik, Institut für Technische Informatik, *BMBF-Förderung von Demonstrationsprojekten für die Funkvernetzung (WLAN) von Hochschulen,* http://wiss.informatik.uni-rostock.de/bmbf/

[13] Adam Stubblefield, John Ioannidis, Aviel D. Rubin, *Using the Fluhrer, Mantin, and Shamir Attack to Break WEP,* AT&T Labs Technical Report TD-4ZCPZZ, Revision 2, August 2001, http://www.cs.rice.edu/~astubble/wep/

[14] Institute of Electrical and Electronics Engineers, Inc. (IEEE), *Port Based Network Access Control,* IEEE draft P802.1X/D11, März 2001, http://www.drizzle.com/~aboba/IEEE/802-1x-d11.pdf

[15] IETF Secure Shell (secsh) working group, http://www.ietf.org/html.charters/secsh-charter.html

[16] Secure Shell Implementierung: OpenSSH, http://www.openssh.org/

[17] IETF Transport Layer Security (TLS) working group, http://www.ietf.org/html.charters/tls-charter.html

[18] OpenSSL Project, http://www.openssl.org/

[19] IETF IP Security Protocol (IPsec) working group, http://www.ietf.org/html.charters/ipsec-charter.html

[20] Mirko Parthey, *IP Security für Linux,* Studienarbeit, http://archiv.tu-chemnitz.de/pub/2001/0008/

[21] William A. Arbaugh, Narendar Shankar, Y. C. Justin Wan, *Your 802.11 Wireless Network has No Clothes,* Department of Computer Science, University of Maryland, März 2001, http://www.cs.umd.edu/~waa/wireless.pdf

[22] Bruce Schneier (Counterpane Systems), Mudge (L0pht Heavy Industries), *Cryptanalysis of Microsoft's PPTP Authentication Extensions (MS-CHAPv2),* http://www.counterpane.com/pptpv2-paper.html

[23] L. Blunk, J. Vollbrecht, *PPP Extensible Authentication Protocol (EAP),* RFC 2284, März 1998

[24] TU Chemnitz, Universitätsrechenzentrum, *Zertifikat Internet-Nutzung,* http://www.tu-chemnitz.de/urz/ZIN/

[25] William R. Cheswick und Steven M. Bellovin, *Firewalls and Internet Security: Repelling the Wily Hacker,* Verlag Addison Wesley, 1994, http://www.wilyhacker.com/

[26] Mark Grennan, *Firewall and Proxy Server HOWTO,* `http://www.grennan.com/Firewall-HOWTO.html`

[27] Paul „Rusty“ Russell, *Linux 2.4 Packet Filtering HOWTO,* `http://netfilter.samba.org/unreliable-guides/packet-filtering-HOWTO/`

[28] Paul „Rusty“ Russell, *Linux 2.4 NAT HOWTO,* `http://netfilter.samba.org/unreliable-guides/NAT-HOWTO/`

[29] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, E. Lear, *Address Allocation for Private Internets,* RFC 1918, Feb. 1996

[30] NCSA HTTPd Development Team, *The CGI Specification,* `http://hoohoo.ncsa.uiuc.edu/cgi/interface.html`

[31] Vipin Samar und Charlie Lai, SunSoft Inc., *Making Login Services Independent of Authentication Technologies,* `http://www.sun.com/software/solaris/pam/pam.external.pdf`

[32] University of Southern California, Information Sciences Institute, Global Operating Systems Technology Group, *The Kerberos Network Authentication Service,* `http://www.kerberos.isi.edu/`

[33] B. Clifford Neuman, Theodore Ts'o, *Kerberos: An Authentication Service for Computer Networks,* `http://www.isi.edu/gost/publications/kerberos-neuman-tso.html`

[34] R. Droms, Bucknell University, *Dynamic Host Configuration Protocol,* RFC 2131, März 1997

[35] Finlayson, Mann, Mogul, Theimer, Stanford University, *A Reverse Address Resolution Protocol,* RFC 903, Juni 1984

[36] Bill Croft (Stanford University), John Gilmore (Sun Microsystems), *Bootstrap Protocol (BOOTP),* RFC 951, Sept. 1985

[37] P. Mockapetris, *Domain names — concepts and facilities,* RFC 1034, Nov. 1987

[38] P. Mockapetris, *Domain names — implementation and specification,* RFC 1035, Nov. 1987

[39] J. Postel, *Internet Control Message Protocol,* RFC 792, Sept. 1981

[40] David C. Plummer, *An Ethernet Address Resolution Protocol,* RFC 826, Nov. 1982

[41] André Breiler, *Differenzierte Bereitstellung von Internetdiensten in öffentlichen Bereichen der Universität,*
`http://archiv.tu-chemnitz.de/pub/2001/0009/`

[42] Internet Software Consortium DHCP,
`http://www.isc.org/products/DHCP/`

[43] Carnegie Mellon University, *DHCP/BOOTP server,*
`ftp://ftp.net.cmu.edu/pub/old/dhcp/dhcp-3.3.7.tar.gz`

[44] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart, *HTTP Authentication: Basic and Digest Access Authentication,* RFC 2617, Juni 1999

[45] Ingo Lütkebohle, *pluggable authentication module for Apache,*
`http://pam.sourceforge.net/mod_auth_pam/`

[46] Thomas Müller, *Installation der AFS-PAM-Module,*
`http://www.tu-chemnitz.de/urz/system/afs/afs_pam.html`

[47] Chris Ulrich, University of California, Riverside (UCR), *(ohne Titel),*
`http://math.ucr.edu:8889/wireless`

[48] Nichole K. Boscia, Derek G. Shaw, NASA Advanced Supercomputing Division, *Wireless Firewall Gateway White Paper,*
`http://www.nas.nasa.gov/Groups/Networks/Projects/Wireless/`

[49] Ralf S. Engelschall, `mod_ssl` F.A.Q. List: *How can I use relative hyperlinks to switch between HTTP and HTTPS?*
`http://www.modssl.org/docs/2.8/ssl_faq.html#ToC23`

[50] Charles E. Perkins, *Mobile IP: Design Principles and Practices,*
Verlag Addison Wesley, 1. Auflage 1997

[51] Helsinki University of Technology, *Dynamics - HUT Mobile IP,*
`http://www.cs.hut.fi/Research/Dynamics/`

Erklärungen

Haftungsausschluß

Der Autor und die Technische Universität Chemnitz übernehmen keine Haftung für Schäden, die sich aus der Benutzung dieser Arbeit oder der zugehörigen Software ergeben.

Schutzrechte

Eingetragene Waren- und Markenzeichen sind im Text nicht als solche gekennzeichnet. Das Fehlen der Kennzeichnung bedeutet nicht, daß diese Zeichen frei verwendbar sind.

Mir sind zum Zeitpunkt der Abgabe keine Ansprüche Dritter an dieser Arbeit und der zugehörigen Software bekannt.

Selbständigkeitserklärung

Hiermit versichere ich, daß ich die vorliegende Arbeit selbständig und ausschließlich mit Hilfe der aufgeführten Quellen angefertigt habe.